Public Transport (2019) 11:341–377
https://doi.org/10.1007/s12469-019-00207-y

ORIGINAL PAPER

Real-world meeting points for shared demand-responsive transportation systems

Paul Czioska[1] · Ronny Kutadinata[2] · Aleksandar Trifunović[3] · Stephan Winter[2] · Monika Sester[1] · Bernhard Friedrich[3]

Accepted: 25 June 2019 / Published online: 29 July 2019

Abstract
While shared demand-responsive transportation (SDRT) systems mostly operate on a door-to-door policy, the usage of meeting points for customer pick-up and drop-off can offer several benefits, such as fewer stops and less total travelled kilometers. Moreover, real-world meeting points offer a possibility to select only feasible and well-defined locations where safe boarding and alighting are possible. This paper investigates the impact of using such meeting points for the SDRT problem with meeting points (SDRT-MP). A three-step procedure is applied to solve the SDRT-MP. Firstly, the customers are clustered into temporary and spatially similar groups and then the alternative meeting points, for boarding and alighting, are determined for each cluster. Finally, a neighbourhood search algorithm is used to obtain the vehicle routes that pass through all the used meeting points while respecting passengers' time constraints. The goal is to examine the differences of a real-world meeting point-based system in contrast to a door-to-door service by a simulation with realistic meeting point locations derived from the map data. Although the average passenger travel time is higher due to increased walking and waiting times, the experiment highlights a reduction of operator resources required to serve all customers.

Keywords Demand-responsive transportation · Shared mobility · Meeting points

✉ Paul Czioska
ikg.paul@czioska.eu

[1] Institute of Cartography and Geoinformatics, Leibniz Universität Hannover, Appelstr. 9a, Hanover, Germany

[2] Department of Infrastructure Engineering, The University of Melbourne, Parkville, VIC 3010, Australia

[3] Institute of Transportation and Urban Engineering, Technische Universität Braunschweig, Hermann-Blenk-Straße 42, Brunswick, Germany

1 Introduction

Demand-responsive transport (DRT) services, also known as Dial-a-Ride, provide a mobility solution based on door-to-door transport on request. They can be operated by companies or statutory authorities, then acting sometimes as part of the public transport. Initially intended as a service with restricted usage (such as for disabled or elderly, sometimes also called *paratransit*), it attracted more general attention in recent years due to emerging mobility solutions and the shortcomings of public transport systems (Nelson et al. 2010; Navidi et al. 2016). The DRT trend has been further boosted by rapid developments in information and communication technologies in the last decade, enabling automatic processing of requests and vehicle assignment.

In contrast to taxicabs, which usually accommodate only one customer (or a customer group) at a time, DRT systems generally focus on larger vehicles, so that multiple passengers can share the ride. In order to distinguish from single customer DRT operators, such as Uber[1] or Lyft[2], the shared mode is often explicitly called a *shared* demand-responsive transport (SDRT) system. The difference is that idle resources are utilized by combining several requests with similar itineraries and time schedules. As a result, a trip is often partly or completely shared with other travellers. Although this induces possible detours to accommodate multiple passengers, the service costs can be reduced and the vehicle capacity utilization can be improved. Growing traffic problems such as congestion or pollution in urban areas encourage a shift towards shared mobility as a more sustainable practice. In the last decade, many service providers have launched new SDRT services, including popular companies such as UberPOOL[3] and Lyft Line[4], and smaller local start-up companies, such as Bridj[5] (Boston, Kansas), Via[6] (New York, Chicago, Washington DC), CleverShuttle[7] (Berlin, Leipzig, München), and Berlkönig[8] (Berlin).

Shared DRT services are also different from shared taxi transportation systems, a mixture between taxicab and bus. Shared taxis mostly operate demand-based without a timetable, but on a fixed or at least partially fixed route. These systems are operating in many countries around the world with different naming, such as *Colectivo*, *Marschrutka*, *Dolmuş* or *Dala dala*, to name a few. The main difference compared to shared demand-responsive transportation systems covered in this paper is that there is no centralized or de-centralized optimization of the routes and the customer assignment to vehicles.

[1] http://www.uber.com
[2] http://www.lyft.com
[3] http://www.uber.com/de/ride/uberpool
[4] http://www.lyft.com/line
[5] http://www.bridj.com
[6] http://ridewithvia.com/
[7] http://clevershuttle.org/
[8] https://www.berlkoenig.de/

Most commercial DRT services offer door-to-door transport, where passengers are picked up and dropped off at their home, or their current location. However, this can lead to considerable detours, since the driver has to stop at many different places and may have to travel into small residential roads. In addition, one-way streets may further extend the trip. A meeting point in the vicinity of a customer's location could reduce the detours required. Furthermore, if the passengers walk a certain distance, they could meet at a place where safe and convenient boarding can be established. In the literature, it was found that people are mostly willing to walk up to 800 m to reach a boarding place (Hess 2012; Millward et al. 2013).

Recently, some well-known SDRT service providers switched to a meeting point-based mode. UberPool and Lyft Line, two big SDRT services, offer so-called "suggested pickup points" which use the fact that a short walk to meeting points allowing them to have an easier way to optimize routes, offer a service for lower fee and save passengers both time and money (Griswold 2017). Also, other service operators, such as Bridj or Via, offer pickup and drop-off at meeting points. If possible, multiple passengers are grouped together and picked up and dropped off at the same location, reducing the number of stops and the necessary service time.

There are several reasons why meeting points can be advantageous for demand-responsive transport services:

– *Safety and convenience* Meeting point locations can be chosen so that safe boarding is possible, without other traffic around and with sufficient parking space, so that luggage can be placed in the vehicle trunk.
– *Identification* By using well-defined meeting points, the driver and the passenger(s) know exactly where to go, and where to find each other. Meeting locations at the doorstep can be ambiguous at times, for example if there are several entrances to a building.
– *Service time* The total service time (including boarding and de-boarding procedure) can be reduced when more than one passenger meet simultaneously, due to a reduction of the total amount of necessary stops.
– *Privacy* The actual origin and destination of customers are not necessarily disclosed, or can be obfuscated (Aïvodji et al. 2016; Goel et al. 2016).
– *Health* The incorporation of walking into the daily transport route can be seen as a contribution to a healthier and more sustainable urban transport. A more sustainable urban planning that includes or encourages walking into a daily transport plan can assist in creating healthier cities (Giles-Corti et al. 2016).
– *Designation* Feasible meeting point locations can optionally be designated by the service operator to make it an official boarding place with reserved boarding areas. In particular, in view of the emergence of autonomous vehicles, it can be beneficial to have locations at hand where a boarding is known to be achievable.

However, the use of meeting points has not gained much attention in research compared to door-to-door DRT systems. Although there are some studies that investigate the impact of meeting points, the actual determination of eligible meeting points in a real city environment has not been done. Often, the Euclidean plane is used for simulation, or all vertices of the street network are considered as potential meeting point location. In reality, however, suitable locations for a safe and convenient pick-up and drop-off are not ubiquitous, since it may not be possible to stop at a junction or in the middle of a street. Moreover, feasible meeting point locations, such as public parking areas, are usually unequally distributed within the city area and dissimilarly reachable by vehicles and pedestrians. Further, the road network may contain obstacles and one-way streets that require large detours to reach some meeting points. Hence, the impacts of these limitations need to be investigated regarding a SDRT scenario.

This paper aims at determining benefits and disadvantages of using real-world meeting points for SDRT-MP systems. A simulation experiment is conducted to demonstrate the impact of different demand levels for supposed SDRT-MP service. The problem is decomposed into multiple sequential problems. Firstly, the incoming requests are clustered based on spatial and temporal similarity of requests. In the second step, real-world meeting points are determined using a GIS approach, which are then used to determine pick-up and drop-off locations for each cluster. During this procedure, it is ensured that each passengers' acceptable walking distances and travel time windows are satisfied. In addition to this, for specific cases, alternative meeting point locations are further identified to potentially allow for better vehicle routing results. Finally, vehicle routing optimization is applied to determine the routes. A similar procedure is applied when passengers are picked up at their doorstep and the two cases are compared to determine if real-world meeting points can be beneficial for SDRT.

2 Literature review

This section provides a literature review of the state-of-the-art research works related to meeting points for demand-responsive transportation systems. Note that the naming of the meeting points is not consistent in the literature. Frequently used denominations include *meeting point*, *pick-up point*, *boarding point*, *stopping point*, *ride-access point* and *rendezvous point*; and correspondingly: *drop-off point*, *deboarding point* and *leaving point*. For the sake of consistency, the denominations *meeting point* (MP) and *divergence point* (DP) will be used in this paper, to emphasize that people share the ride in between.

The majority of previous works on meeting points consider the standard ride-sharing case with each driver having a unique origin and destination and not acting as a service provider. The works considering this case cover several different research areas, such as the optimization of the inner-city ride-sharing (Aissat and Oulamara 2014, 2015; Balardino and Santos 2016; Stiglic et al. 2015; Correa et al. 2017), the optimization of long-distance ride-sharing (Czioska et al. 2017), the optimization of carpooling for a fixed community (Chen et al. 2016), or privacy protection (Aïvodji et al. 2016; Goel et al. 2016). Alternatively, some optimize the placement of meeting points (Goel et al. 2017) or the cartographic representation of meeting points (Rigby et al. 2013; Rigby and Winter 2016; Rigby et al. 2016). Usage of meeting points in demand-responsive context was so far investigated only by Häll et al. (2008) and Martínez et al. (2014), but the usage of real-world meeting points for demand-responsive transportation is, to the best of the authors' knowledge, currently not investigated.

These works use various methods to determine the meeting point set. A simple approach is to place the meeting points randomly in the investigation area, without considering the underlying street network. This can be done a priori to limit the set of feasible places (Stiglic et al. 2015) or a posteriori, when the group of travellers is already known (Martínez et al. 2014). Another possibility is to use all nodes in the street network graph (Aissat and Oulamara 2014, 2015; Balardino and Santos 2016; Aïvodji et al. 2016). Another example is a study by Goel et al. (2016), where a randomly chosen subset of major street intersections is used as the meeting point set. In the study of Häll et al. (2008), a hybrid method is applied: a regular grid of meeting points is initially created, and those that are not accessible using the road network are removed. However, it is important to consider real-world meeting points derived from map data since this narrows down available meeting points and introduces additional constraints to the problem. Currently, map-based meeting point determination is considered only for ride-sharing by Correa et al. (2017); Czioska et al. (2017) and for carpooling by Chen et al. (2016).

Besides determining the available meeting point set, it is also important to consider the method to select which meeting point to be used. A simple method is to perform some optimization based on Euclidean distance (Stiglic et al. 2015). If the group of customers sharing a ride is already known, another approach is to use the centroids of customer group locations as meeting and divergence points. However, they are hence not necessarily aligned with the street network or restricted to a predefined subset of candidates (Martínez et al. 2014). Furthermore, the selection process also needs to consider the walking distance imposed onto the travellers. Unlike in the study by Correa et al. (2017) (where customers drive to the meeting points) and by Aissat and Oulamara (2014), most other approaches limit the walking distance, either during the ride sharing matching phase Stiglic et al. (2015) or the passenger clustering phase for SDRT (Martínez et al. 2014). However, the walking distance constraint may not be explicitly predetermined since the approaches by Stiglic et al. (2015); Martínez et al. (2014) do not consider the underlying street network, as previously mentioned.

While most approaches focus on selecting a single meeting point for each stop, a methodology has been proposed that uses lines or areas as potential meeting zones (Rigby et al. 2013, 2016), leaving the exact meeting point choice to the travellers. In a study by Rigby et al. (2013), an opportunistic client user interface technique called *launch pads* was developed, showing passengers the area in which they could potentially be picked up by a driver within a certain time interval. Later, the launch pads have been extended to a continuous representation of vehicle accessibility (Rigby et al. 2016). However, it is noteworthy that the study by Häll et al. (2008) has discovered that allowing the customers to choose the meeting point may lead to an inefficient use of meeting points. For instance, due to the very dense network of meeting points in the study by Häll et al. (2008), each customer is able to select the closest meeting point such that the walking distance from the customer's doorsteps is never large, which yields a similar result to its door-to-door counterpart.

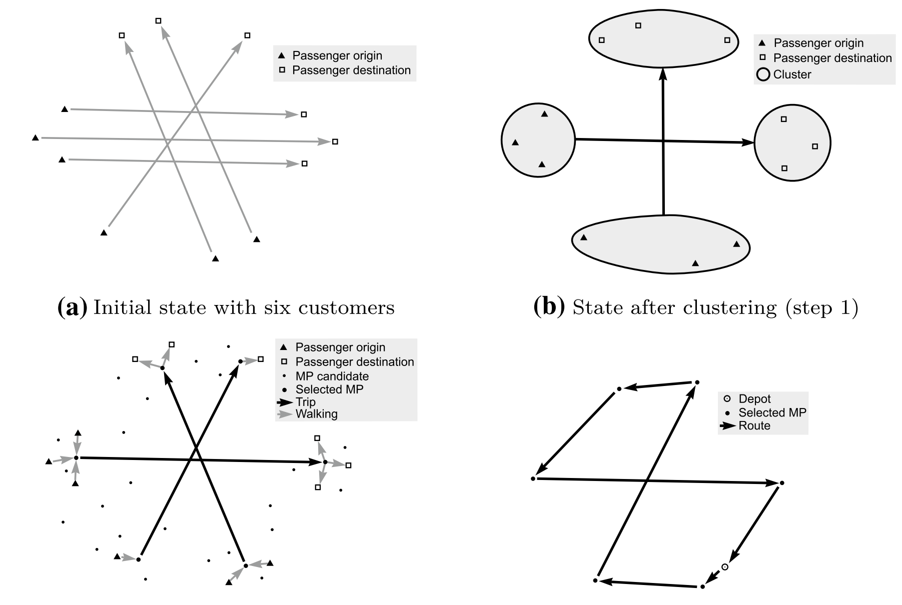
Contrary to the study by Häll et al. (2008), other works have shown more positive results in using meeting points. In a ride sharing scenario, where a driver is only allowed one pick-up and drop-off (yet may accommodate more than one passenger), a computational experiment (Stiglic et al. 2015) has demonstrated that the use of meeting points may improve the number of matched participants (up to 6.8%) as well as mileage savings (up to 2.2%), depending on the number of available meeting points. In an urban SDRT service that utilises meeting points (Martínez et al. 2014), such service may replace 53% of private car trips longer than 5 km performed during the morning peak (each minibus replaces, on average, 9.15 private cars) while still maintaining some financial profitability. Note that no comparison with door-to-door service is made in this study by Martínez et al. (2014). However, obviously the average trip time of customers using the service (both ride sharing and SDRT setting) experience increased total trip time due to detour and walking. It is also reported in a study by Mageean and Nelson (2003), that the level of meeting points dispersion affects operational complexity and user satisfaction. Hence using artificial meeting points, as it was done in previous works, to evaluate SDRT-MP service does not guarantee that benefits and drawbacks will be evaluated appropriately.

A particular interest is given to the study by Martínez et al. (2014) that uses a hierarchical clustering approach to group the passenger requests. The clustering of passengers takes into account normalized values of the distance between trip origins and destinations, and the difference in arrival time. It is furthermore constrained to a maximum distance radius and diameter, a maximum time, and a maximum cluster size. These clusters are then subsequently combined to vehicle routes by Binary Integer Programming.

In light of this review, this paper presents an investigation of the use of real-world meeting points based on map data for SDRT service. The contributions of this paper are threefold. Firstly, this work sought to confirm the benefits of using meeting points when constrained to only practical stopping locations on the road network, such as parking lots or service stations. This strategy is opposed to more basic approaches, e.g. using the spatial centroid of requests, as applied in the work by Martínez et al. (2014). Secondly, the proposed methodology of this paper explicitly constrains the walking distance of the travellers since it uses a clustering step that considers the distance based on the actual street network, unlike those methods used by Stiglic et al. (2015) and Martínez et al. (2014). The explicit constraint allows to limit the walking distance to a previously-proven acceptable amount (Hess 2012; Millward et al. 2013). Finally, this paper serves as a case study of the implementation of SDRT-MP to the city of Braunschweig, Germany. This demonstrates the potential of the SDRT-MP in a real-life scenario that extends beyond hypothetical networks and demands.

3 Methods

In this section, the methods to assign passengers to meeting points and to create vehicle routes based on these points are described. The workflow consists of three discrete steps:



(a) Initial state with six customers

(b) State after clustering (step 1)

(c) State after MP candidates selection (step 2)

(d) State after vehicle optimization (step 3)

Fig. 1 Illustration of the outputs of the clustering, MP selection and route optimization steps within the workflow

1. Clustering
2. MP candidates selection
3. Route optimization

Figures 1 and 2 visualize the processing chain. In a nutshell, the demand (Fig. 1a) is initially clustered into groups with similar origins/destinations and time schedules (Fig. 1b). Secondly, each group is separately split into *trips*, with each trip having a common meeting and divergence point and potential alternative meeting points (Fig. 1c). Finally, the vehicle routing problem is solved to construct the routes (Fig. 1d).

The workflow is achieved by executing five algorithms and elementary processes. Figure 2 shows the interconnections and data exchange among the algorithms. The execution starts with the raw demand, which is initially clustered by Algorithm 1, yielding clusters of requests. These clusters are subsequently used as input for the Algorithms 2 and 3. For each cluster, these algorithms determine *trips*, each having one or multiple meeting and divergence points. These points are reachable from all passengers of the cluster. Consecutively, the trips are again clustered by Algorithm 1, which is done just to enable a parallel processing of the route optimization. Details of the optimization can be accessed in Sect. D.1. The output of the optimization is a set of routes for each trip cluster. The final step is then to combine all the routes from the different clusters into longer routes to lower the number of necessary

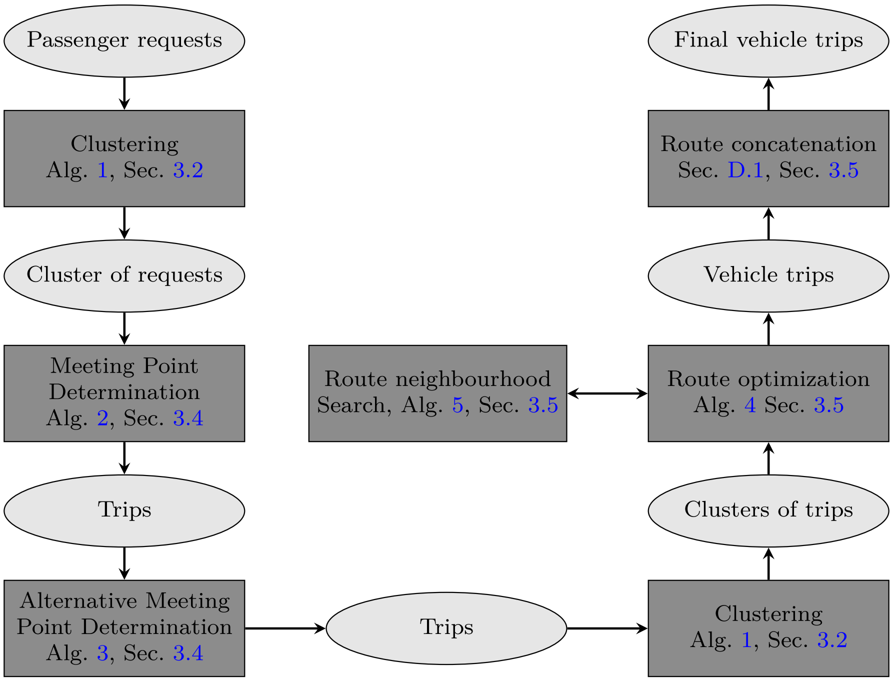


Fig. 2 Overview of algorithms used and data exchange in the workflow

vehicles. This is done by Algorithm 4, which calls Algorithm 5, yielding the final vehicle routes.

In the following, the data prerequisites and the details of each step of the workflow as well as the algorithms are described.

3.1 Data prerequisites

The approach used in this work needs the following input and parameters:

– *Street network* A network graph $G = (V, E)$ consisting of vertices V and directed edges E. Each edge $e \in E$ has a distance $d(e)$, and a corresponding vehicle driving time $t^{\mathrm{driv}}(e)$ and/or a corresponding passenger walking time $t^{\mathrm{walk}}(e)$, depending on the type of an edge e.g. footpaths have no defined vehicle driving times.
– *Meeting point candidates* A set of meeting point candidates $\mu \in M \subset V$ and a set of potential divergence point candidates $\delta \in D$ are created using the methodology described in Sect. 3.3. In this paper, these two sets are considered as equal, so that $M = D$.
– *Demand* A set of passenger requests P, where each passenger $\rho \in P$ defines an origin vertex v^+, a desired destination vertex v^-, and a desired departure time t^+.

Depending on the maximum passenger walking distance d_*^{walk}, each passenger $\rho \in P$ is able to walk to/from a set of reachable meeting points M_ρ and divergence points D_ρ, respectively. Furthermore, the allowed detour time for a passenger $t_{*\rho}^{\text{detr}}$ is limited by either a hard time threshold that cannot be exceeded (t_*^{detr}) or the travel time multiplied with a ratio value r_*^{detr}, which prohibits exceeding detours on long trips:

$$t_{*\rho}^{\text{detr}} = \min\left(t^{\text{driv}}\left(v_\rho^+ \to v_\rho^-\right) + t_*^{\text{detr}}, t^{\text{driv}}\left(v_\rho^+ \to v_\rho^-\right) \cdot r_*^{\text{detr}}\right) \tag{1}$$

This personalized maximum detour time can then be inserted in the calculation of the latest acceptable arrival time at the destination t_ρ^-:

$$t_\rho^- = t_\rho^+ + t_*^{\text{wait}} + t_*^{\text{serv}} + t_{*\rho}^{\text{detr}}. \tag{2}$$

where t_*^{wait} is the maximum passenger waiting time, and t_*^{serv} is the vehicle serving time (boarding/alighting).

These and other parameters used in the experiment are presented in Table 1.

3.2 Clustering

The clustering method in this paper is used to join similar transport requests into clusters that have a fixed size. It utilises an idea from the field of data analysis, where abstract points in space are classified using various distance measures in multidimensional space. The method is greedy and produces clusters that will have the same size with the possible exception of the last cluster.

The proposed approach in this paper applies the spatio-temporal constraint checking during the MP candidates selection step (Sect. 3.4) to use actual distances based on the street network instead of Euclidean distance, which is used during the clustering. Fixed size clusters are necessary to avoid combinatorial explosion during assignment of passengers to MP, since the method uses set partitioning, which is NP-complete.

Similarly to many classification algorithms (Hastie et al. 2009), we consider a vector of features $\mathbf{x} = [x_1, x_2, \dots, x_N]$, which in this case are the features of a transport request: origin vertex coordinates, destination vertex coordinates and desired departure time i.e. $\mathbf{x} = [v^+, v^-, t^+]$. The cost function is defined as the Euclidean distance: $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$, where x_i is the ith element of a feature vector $\mathbf{x}$. Note that units for space and time are in meters and seconds, which have the same magnitude in this work, hence neither weighting nor normalization was performed.

Consider a cluster C that is a set of transport requests. Then, the distance of a new transport request $\mathbf{x}$ from cluster C is defined as:

$$d(C, \mathbf{x}) := \sum_{\mathbf{c} \in C} d(\mathbf{c}, \mathbf{x}), \text{ where } C \neq \emptyset. \tag{3}$$

The maximum number of requests for one cluster is fixed, hence the created groups are equally sized (except, potentially, the last one). Clustering works iteratively, it processes one request after another. To gain better understanding of this, consider a matrix X where the rows are transport requests i.e. $\mathbf{x} \in X$. In the beginning of the clustering procedure we set $C = \mathbf{x}_0$ i.e. the first request from X and remove $\mathbf{x}_0$ from X. In the next step, the cost from the cluster C to all other vectors is calculated using Eq. (3) and the vector that has the smallest cost is added to C. This is repeated until the maximum allowed size of a group is reached in C, after which the whole process is repeated again until all requests are processed. Since the method is greedy and is mainly influenced by the first request, some clusters can end up with the first request being much further away from the rest. This problem is corrected during the assignment of passengers to meeting points which will split such clusters into two or more clusters.

The clustering algorithm is presented in the Appendix B (Algorithm 1). It is a novel clustering algorithm for creating fixed size clusters. The main difference from approaches found in the literature (e.g. Najmi et al. 2017) is that our approach requires only the maximum size of a cluster to be specified, whereas other approaches require determining the number of clusters in advance and have no guarantees on the number of transport requests per cluster. Therefore, our clustering approach is better suited to be used as a preprocessing step for algorithms that have high computational complexity, such as the ones presented in Sect. 3.4, since the complexity can be controlled. The complexity of the clustering algorithm, in time, is $\mathcal{O}(n^2)$ which is very competitive to the algorithms based on k-means which are commonly used for the same task. The derivation of the algorithm complexity is presented in Appendix B.1.

3.3 Meeting point candidates generation

Feasible MP candidates $\mu \in M$ (consequently, also DP candidates) are identified automatically based on real-world map data. For this study, we used data from OpenStreetMap[9] and identified feasible locations by the tags of the map features. In order to ensure safety and convenience aspects like boarding places with reduced traffic, possibilities to park and easily recognizable places, the candidate locations are limited to the following selection:

- publicly accessible parking places without parking fees,
- side road intersections (with all adjacent roads having a maximum speed of $\leq$ 30 km/h),
- turning areas (mostly at the end of a cul-de-sac),
- petrol stations.

In practice, frequently used meeting places also include the curb or public transport stops. However, this possibility is intentionally disregarded, as it would be irresponsible to encourage people to meet in this manner. Further, it is not advisable according to road traffic regulations in most countries.

[9] http://www.openstreetmap.org ©*OpenStreetMap Contributors*

If parking areas and petrol station areas are originally mapped as polygon features in OpenStreetMap, they are first converted to point features using the centroid. Each candidate location is connected to the street network G with an edge between the meeting point location and the closest point on the closest edge. If the closest edge is not reachable by vehicles (such as a footpath), a second edge is inserted at the closest drivable edge. Also, if the closest edge is not accessible on foot, another edge is inserted at the closest walkable edge, so that both pedestrians and drivers can reach the meeting points. If a meeting point is not reachable for drivers or passengers (e.g. located on private ground), it is removed from the set.

3.4 Meeting point candidates selection

In this step, feasible meeting and divergence points are assigned to the previously determined clusters. Most likely, not all customers of a cluster will be feasible for a single meeting and divergence point, so the group has to be split up into subgroups which can reach one (or more) common meeting and divergence point(s). Such a subgroup is called a *trip* τ. It is defined as a tuple consisting of one or more passengers with one or more meeting and divergence point candidates, each associated with a corresponding time window, indicating the valid boarding and alighting times: $\tau_i = \big(\{\rho_{i,1}, \rho_{i,2}, \cdots\}, \{\mu_{i,1}, \mu_{i,2}, \cdots\}, \{\delta_{i,1}, \delta_{i,2}, \cdots\}\big)$ such that $\forall \mu,\ \exists\, [t^E(\mu), t^L(\mu)]$ and $\forall \delta,\ \exists\, [t^E(\delta), t^L(\delta)]$.

A trip τ is considered feasible if the time windows and walking distances are within the thresholds for all customers P_τ of the trip. The earliest (E) and latest (L) pick-up times for a MP μ are determined by:

$$t^E(\mu) = \max_{\rho \in P_\tau} \left(t^+_\rho + t^{\text{walk}}(v^+_\rho \to \mu) \right), \tag{4}$$

$$t^L(\mu) = \min_{\rho \in P_\tau} \left(t^+_\rho + t^{\text{walk}}(v^+_\rho \to \mu) + t^{\text{wait}}_* \right). \tag{5}$$

The earliest and the latest drop-off time, t^E and t^L, for a DP δ are determined by:

$$t^E(\delta) = \max_{\rho \in P_\tau} \left(t^+_\rho + t^{\text{driv}}(v^+_\rho \to v^-_\rho) - t^{\text{walk}}(v^-_\rho \to \delta) - t^{\text{serv}}_* \right), \tag{6}$$

$$t^L(\delta) = \min_{\rho \in P_\tau} \left(t^+_\rho + t^{\text{wait}}_* + t^{\text{detr}}_{*\rho} - t^{\text{walk}}\left(v^-_\rho \to \delta \right) - t^{\text{serv}}_* \right). \tag{7}$$

3.4.1 Single MP/DP selection

The task is to split up the groups coming from the clustering procedure as efficiently as possible into subgroups, such that the group size of the subgroups remains as large as possible while satisfying constraints coming from the underlying real-world map. This problem can be defined as *set partitioning problem*, with the objective to find the smallest number of subgroups that satisfy the walking and time constraints

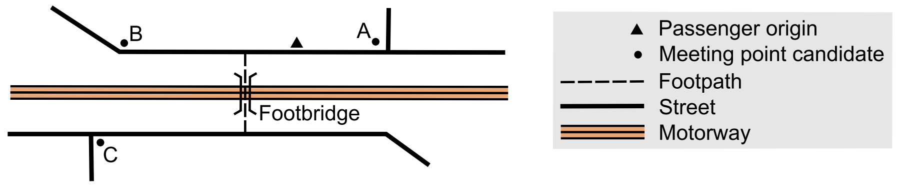


Fig. 3 Schematic drawing of a situation with useful alternative meeting point search

for each passenger from a given cluster. Since the set partitioning problem is NP-complete, the input size is naturally limited. However, due to the fixed cluster size resulting from step 3, we propose a novel recursive dynamic programming algorithm that yields the optimal solution for given clusters.

Initially, the algorithm attempts to put all customers within a cluster into a trip with a common meeting and divergence point. If this is spatially and/or temporally infeasible, the group is split into all possible subgroup combinations, and their feasibility is checked likewise. This is done recursively for each subgroup until a feasible solution is found. The main objective is to split the initial cluster into as few separated trips as possible. If there are multiple different combinations with the same number of necessary trips, the combination with the least sum of squared walking distances of all customers is chosen. The squaring is applied to penalize longer walking distances more than short distances, so that the walking distances have low variation.

By storing the intermediate results during the recursive process the computation time can be considerably reduced (*dynamic programming* approach). Thus it is possible to process reasonably sized clusters within an acceptable time. The proposed MP candidates selection method is outlined in Appendix C (Algorithm 2).

3.4.2 Alternative MP/DP selection

The previously described algorithm returns exactly one MP and DP (including time windows) for each trip. However, there can be other meeting points worth considering from the operators perspective. For instance, consider the scenario shown in Fig. 3. In this case, the closest meeting point from the passenger origin (Point A) is chosen, which is located north of the motorway. Two other meeting point candidates, namely B and C, would also be feasible for the trip, but they have not been chosen because of longer walking distances. For the operator, however, it could be advantageous in the routing phase to also consider C, since it offers the possibility to approach the passenger from the south without having to take a large detour around the motorway. From the passenger's perspective, it is only a minor extension of the walking path via the footbridge.

To combat this, a second algorithm is proposed to identify alternative meeting points, which can be considered during the route optimization phase. Note that only one meeting point is still served by the vehicle, but the alternative meeting points provide more options for creating shorter routes. This extension is, to the best of our knowledge, unique among the recent literature concerning meeting points and provides advanced optimization possibilities for the operators.

Using a *shortcut ratio threshold* σ, an alternative meeting point candidate μ_a is considered if

$$\frac{t^{\text{driv}}(\mu \rightarrow \mu_a)}{t^{\text{walk}}(\mu \rightarrow \mu_a)} \geq \sigma, \tag{8}$$

where μ is the initially identified meeting point candidate. In the example shown in Fig. 3, Point C is likely to satisfy Eq. (8), whereas Point B is unlikely. A larger σ indicates fewer alternatives, which helps to reduce the input size of the vehicle routing problem. For faster computation times, all ratio values between all meeting point pairs can be precomputed and stored. The proposed algorithm is outlined in Appendix C (Algorithm 3). It recursively checks the next best option until no further useful alternative can be found. Also, note that the corresponding time windows of the alternative meeting points have to be checked.

3.5 Route optimization with final meeting points selection

In this step, the trips having one or more meeting point alternatives resulting from the previous unit are combined to create the vehicle routes. In theory, this vehicle routing problem can be solved with any optimization method that is able to solve this class of problems, with minor adaptation for this use case. In this project, a neighbourhood search approach is used. In order to speed up the process, a second clustering is initially applied on the trips to form equally-sized trip clusters with similar itineraries, which can then be solved in parallel. For this, the already described clustering method (Sect. 3.2) is reused; however, with a different threshold (k_2). Certainly, this makes it necessary to append a postprocessing step to combine the results of the simultaneously derived vehicle routes (Sect. 3.5).

The proposed route optimization problem itself differs from those in the literature mainly because of the alternative meeting points for each boarding/alighting procedure that introduces an extra complexity to the problem. The presented approach of the route optimization problem extends the formulation by Kutadinata et al. (2019), which is similar to typical vehicle routing problems (Cordeau and Laporte 2007; Ropke and Pisinger 2006; Parragh et al. 2008; Li et al. 2014) that use directed graph modelling. The novelty of the formulation used in this work stems from the fact that it does not introduce extra nodes to model the alternative meeting points. Instead, they are incorporated as decision variables and the travel times, distance, and time-windows are functions of these decision variables. The benefit of this approach is that it requires very little modifications to the formulation that are already presented in the literature. As a result, most existing heuristics can be used to solve the problem. For the sake of brevity, the following summarises the optimization problem, while the complete formulation is presented in the Appendix D.1.

The problem is formulated as an optimization model on a directed graph, which is a different graph to the previously defined directed street graph G. Firstly, the

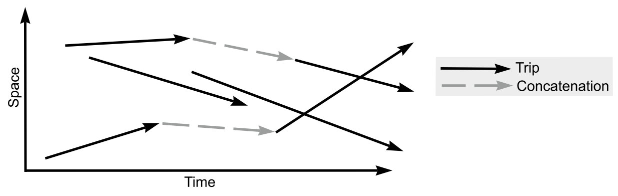


Fig. 4 Concatenation of routes to form longer ones

stops to be visited are represented by a set of nodes, which include the vehicle starting points, the stops to be visited (both pick-up and drop-off points), and a depot. Secondly, although each stop can have multiple meeting points, it is still represented by a single node. The meeting points for each node are represented as a set of decision variables to be optimized. Furthermore, each node is associated with several parameters that define the trip demand, namely: the number of passengers at the stop, the pick-up and drop-off time windows. Similarly, each directed edge is associated with distance and travel time parameters. Due to the multiple alternative meeting points, these parameters, except the number of passengers, are functions of the chosen meeting points.

The optimization objective function is a trade-off between the “service level cost” and the operating cost. The service level cost takes into account the passengers late time, pick-up wait time, and detour time, whereas the operating cost considers the total number of vehicles used and the total mileage. Furthermore, note that the time windows are treated as soft constraints that necessitate the use of penalties for late pick-up and drop-off in the objective function (as part of the service level cost). This allows the optimization algorithm to choose a solution that has late services that are justified by the savings in other aspects. Typically, higher penalty weight parameters are used to avoid an unreasonable number of late arrivals. However, if hard time window constraints are desired, simply choose very large penalty values.

A two-layer neighbourhood search algorithm is used to solve the problem. The top layer optimizes the trip allocation to vehicles, which repeatedly calls the bottom layer that optimizes the route of each vehicle including the meeting and divergence point selection. The algorithms are presented in the Appendix D (Algorithms 5 and 4).

The output of a single optimization process described above is a group of routes, each route performed by a single vehicle. Since the route optimization for each cluster of trips is performed in parallel, some of the routes can now be *concatenated* to reduce the total number of routes (and consequently the total number of vehicles used). Thus, this step can be described as a problem of maximizing the number of concatenations and is formulated and solved using a Linear Programming (LP). To ensure that a concatenated route can still be feasibly served by a vehicle, a constraint is applied to ensure that there is enough time to travel from the last stop of a precedent route to the first stop of the subsequent one. The detailed formulation of the route concatenation is presented in the Appendix D.2.

The concatenation solution is presented as an illustrated space-time diagram in Fig. 4. Note that the final statistics presented in this paper are the output of the LP problem.

3.6 Dynamic scenario considerations

The problem solved in this paper is static; therefore, the dynamic problem is not in the focus of this paper. However, the workflow described in this section can also be used to handle dynamic requests, either through an event-driven approach or periodic optimization, i.e. a rolling horizon approach (Najmi et al. 2017; Eser et al. 2018). Firstly, once a new request arrives, the clustering algorithm can be adapted to insert the new request into the closest non-full cluster. Recall that there will be some non-full clusters due to the splitting done during the meeting point selection. Secondly, due to the decentralised nature of the subsequent processes, the meeting point selection and the route optimization can be done only to the affected clusters. Finally, the route concatenation needs to be executed again only if the last stop of the affected cluster changes. However, as previously mentioned, addressing the dynamic situation is not the focus of this paper and is hence reserved for future work.

4 Simulation experiment

To evaluate the potential benefits of the use of meeting points, a simulation experiment is carried out, comparing the performance of a meeting point-based service (MP) with a door-to-door service (DS) used as a baseline scenario. In the MP service, all steps described in Sect. 3 are executed, whereas the DS service only uses the vehicle routing optimization, applied on the raw demand data. As a result, this simulation focuses on investigating the benefits of the meeting point determination rather than the route optimization process. Note that for the simulation experiments, the parallel version of the route optimization is used in order to speed up the computation and solve the instances within a reasonable CPU time. Thus, the same heuristics are used to solve both the MP and DS cases to ensure a fair comparison.

4.1 Simulation setup

The simulation experiment investigates the impact of using meeting points for various demand volumes. It is expected that, as the demand increases, the impact of using meeting points will be more significant. A total of 10,000 passengers are randomly generated based on the procedure described in Sect. 4.5 and subdivided into four demand instances (1000, 4000, 7000 and 10,000). Each demand instance is evaluated in both the MP and DS scenarios. To simplify the experiment, only a static problem is considered, i.e. all trips are known in advance. The parameters used in the optimization and meeting point algorithm are shown in Table 1.

The maximum cluster size for the initial clustering (k_1) is a hyperparameter that can be tuned in order to balance the trade-off between the quality of the result and

Table 1 Simulation parameters

Parameter	Unit	Description	Value used for simulation
t_*^{wait}	s	Maximum passenger waiting time	1200 s
d_*^{walk}	m	Maximum passenger walking distance	800 m
t_*^{detr}	s	Maximum allowed vehicle detour time	1200 s
r_*^{detr}	%	Maximum allowed vehicle detour time	25%
t_*^{serv}	s	Vehicle service time (for boarding/alighting)	120 s
σ	%	Shortcut ratio threshold (cf. Sect. 3.4)	50 %
k_1	–	Maximum cluster size for the initial clustering	11 passengers
k_2	–	Maximum cluster size for the re-clustering	10 trips
q_*	–	Maximum vehicle capacity	9 passengers
c_*^{dist}	km^{-1}	Vehicle distance dependent cost	$1\ \text{km}^{-1}$
c_*^{vehi}	–	Vehicle capital cost	2000
c_*^{wait}	s^{-1}	Passenger wait time cost	$0.5\ \text{s}^{-1}$
c_*^{late}	s^{-1}	Passenger late time cost	$5\ \text{s}^{-1}$
α	–	Passenger wait time cost growth	0.5
β	–	Passenger late time cost growth	2

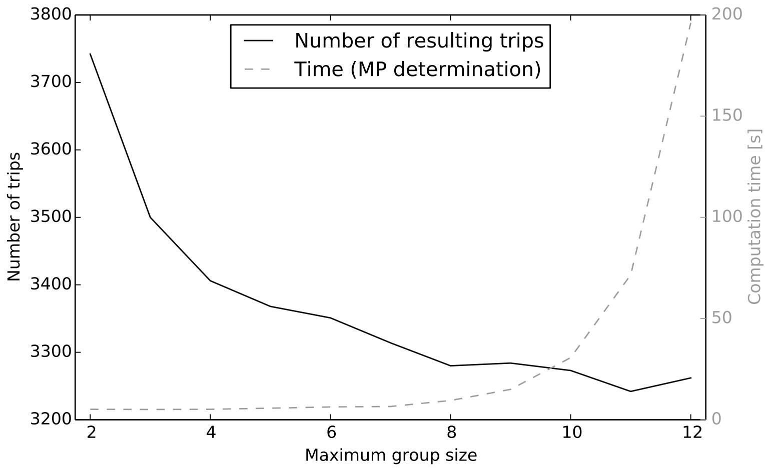


Fig. 5 Cluster size influence on the computational time (5000 passengers)

the computation time. Larger clusters lead to better results in terms of lowering the necessary boarding stops, since the meeting point determination method yields the optimal solution for a given group. On the other hand, the computation time grows exponentially with increasing cluster size. Figure 5 shows a sensivity analysis to determine a reasonable value. The maximum cluster size varies from 2 to 12 and the resulting number of trips and the computation times are recorded. The experiment was performed by a Python 3 script using parallel processing on a 4th generation desktop orientated Intel i5 processor with 4 cores. As it can be seen, a larger input cluster size leads to fewer total number of trips, indicating a solution of higher quality. But, on the other hand, the computation time grows very quickly. In the simulation experiments, a maximum cluster size of 11 was chosen, to obtain good results within a reasonable computation time.

Similarly, for the re-clustering (k_2), a value of 10 was determined as providing a good trade-off between calculation time and result quality.

4.2 Implementation details

Demand generation, clustering and meeting point determination are implemented and tested in Python 3.6.5. Demand generation and meeting point determination perform shortest path computations by using pgRouting, which is an extension of the PostgreSQL database. This is done by establishing a connection with a database, located on a remote server, and executing SQL queries from Python. In the Python code several well established open source libraries are used: numpy (for linear algebra computations), psycopg2 (for SQL connection), h5py (for storing the data in HDF5 format), matplotlib (for plotting the results). Python code is parallelized with multiprocessing module provided by the official Python distribution that comes with Ubuntu 18.04.

Vehicle route optimization is implemented in MATLAB. Parallel processing was set up using MATLAB Distributed Computing Server that solves the routing problem of each cluster independently. Once the optimization of all clusters was completed, the route concatenation code was then executed without parallelization.

4.3 Street network

For the simulation experiment, the city of Braunschweig, Germany, is used as a spatial template. It is a medium-sized city with 250,000 inhabitants and a typical European city structure: the centre is dominated by its historical core with an irregular street network and pedestrian precincts, surrounded by a densely populated area with a more regular street network. In the outskirts, the population density is significantly lower and there are some industrial areas.

The street network has been obtained from OpenStreetMap and transformed into a routing-enabled graph (Fig. 6). As previously mentioned in Sect. 3.1, vehicle driving times $t^{\text{driv}}(v_i \to v_j)$, passenger walking times $t^{\text{walk}}(v_i \to v_j)$ are calculated for every edge $e \in E$. The driving time depends on the maximum allowed speed. The walking time t_e^{walk} is determined by assuming a constant walking speed of 4.8 km/h, using the mean speed for active-transport walking trips revealed by Millward et al. (2013). The traversal of footpaths, cycle ways and stairways is prohibited for vehicles. Likewise, pedestrians are not allowed to walk on major roads or motorways.

As potential origins, residential buildings from a “Level-of-Detail 1” building model of the city of Braunschweig (Municipality of Braunschweig 2016) with a size of more than $100\,m^2$ are added to the network. Likewise, workplace buildings are used for potential destinations. All considered buildings are connected to the street

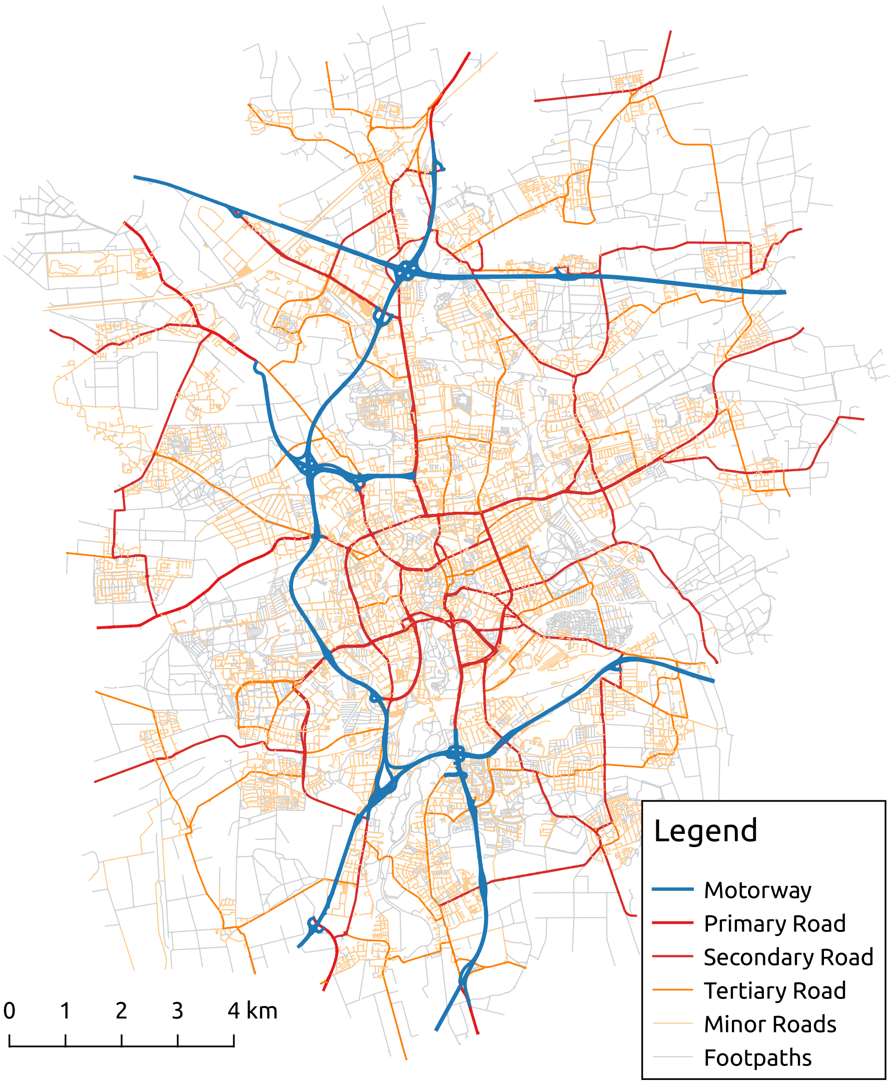


Fig. 6 Routing network (Source: OpenStreetMap)

network to model the whole path of the user (Fig. 7). In total, the network consists of 88,381 nodes and 99,497 edges, including 26,845 potential home and 2615 potential work locations.

4.4 Braunschweig meeting points

Using the methodology described in Sect. 3.3, 3475 meeting points have been detected within the investigation area, based on OpenStreetMap data. Slightly more than 50% of the meeting points are street intersections that can be found in low

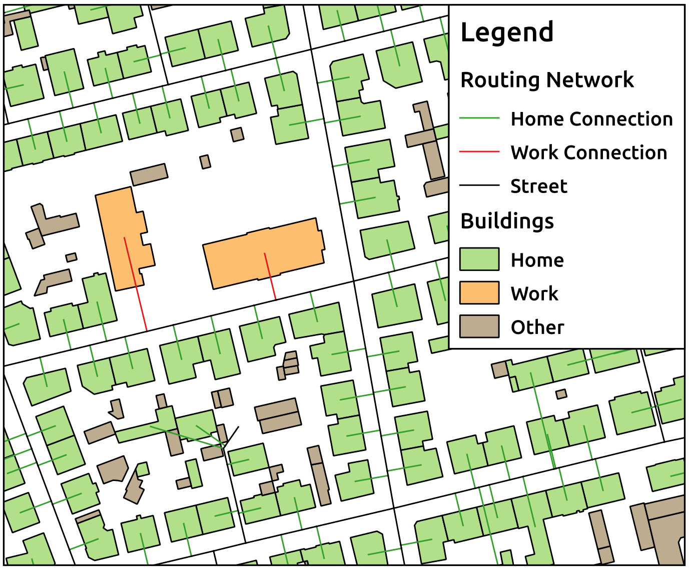


Fig. 7 Connected origins and destinations

speed areas, and approximately 20% are parking places and turning areas. Only very few meeting points are located at petrol stations. In relation to the total investigation area size of 193 km^2, the overall MP density is approximately 18 per km^2, and in dense urban areas up to 40 MP per km^2. The observed mean distance to the nearest neighbouring MP is approximately 70 m.

Compared to the total number of nodes in the graph with approximately 30,000 nodes, the number of candidate locations is relatively low, which is in contrast to other works that use all street intersections as candidate locations (e.g. Balardino and Santos 2016). A lower amount of suitable meeting points better represents the reality, since not every intersection is feasible for safe boarding. Hence, we expect the results to better reflect the actual situation.

4.5 Demand generation

A set of randomly generated passenger requests P is used, defining for each passenger $\rho \in P$ an origin node v_ρ^+, a desired destination node v_ρ^-, and a desired departure time t_ρ^+. For the DS case, the closest node that is reachable by vehicles is stored as the origin, and for the destination likewise. The spatial distribution of the trip requests is based on a weighted sampling of building data. Specifically, the probability of a building to be chosen as origin or destination is proportional to its volume. To prevent huge factory buildings to be chosen disproportionately often, the probability of a building being selected is capped equivalently to having a volume of $10,000\,\mathrm{m}^3$. Figures 8 and 9 visualize the spatial distribution of customers within the investigation area.

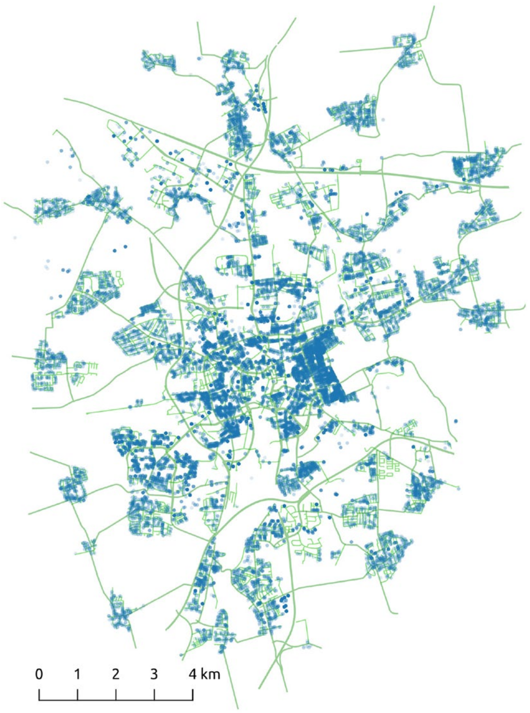


Fig. 8 Customer origin locations

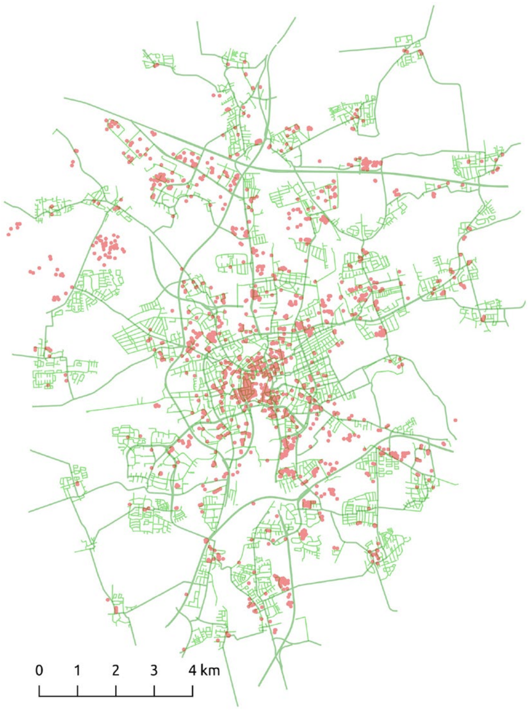


Fig. 9 Customer destination locations

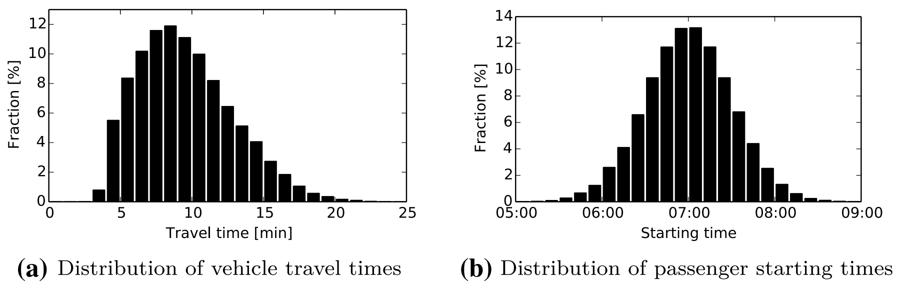


Fig. 10 Statistics about randomly generated customer demand

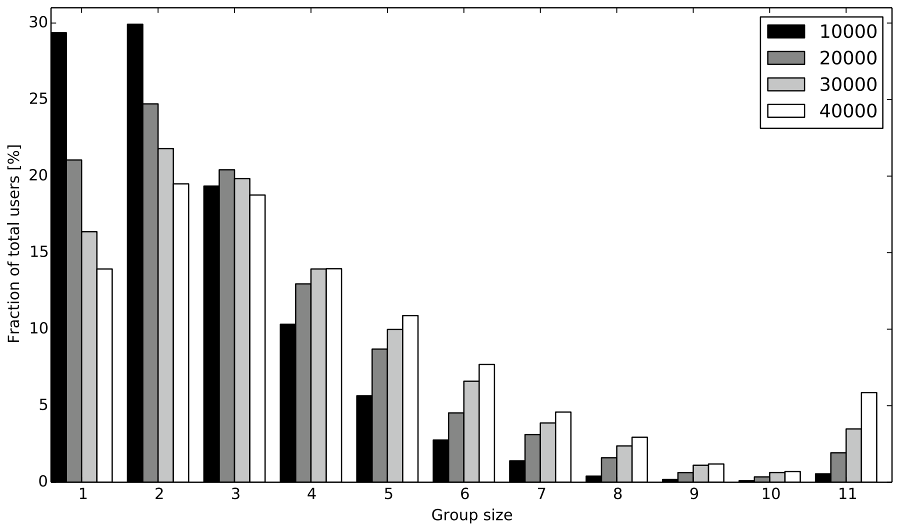
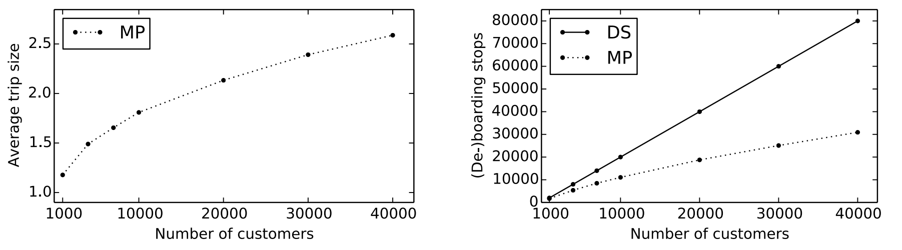


Fig. 11 Fraction of users per group size after MP candidates selection for different total demand scenarios

Furthermore, trip requests having their destination within 2000 m Euclidean distance of the origin are avoided since the passengers are assumed to walk or cycle. Figure 10a shows the resulting distribution of direct travel times via driving between origins and destinations.

In order to simulate a busy morning commute peak, the temporal distribution of the requests follows a Gaussian distribution centred at 07:00 AM with a standard deviation of 30 min (Fig. 10b). Moreover, as there are no actual depots in this problem, the depot location is chosen as the node which has the smallest maximum travel time to the other nodes in the network graph.



(a) Average customers per pick-up **(b)** Amount of boarding and alighting stops

Fig. 12 Comparison between meeting point and doorstep simulation concerning trip size. *DS* Doorstep, *MP* meeting points

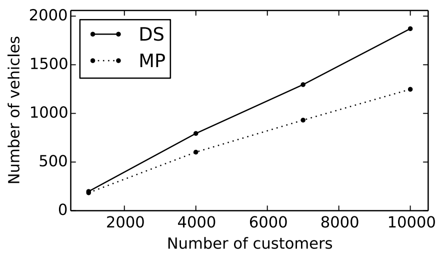
5 Results

Figure 11 shows the group size histogram after the MP candidates selection step for four cases with different total number of customers, up to 40,000. While after the initial clustering phase nearly all groups have the maximum allowed size of 11 passengers, the next step splits these groups into feasible subgroups. The result naturally correlates with the average number of customers per pick-up (Fig. 12a). Generally, it can be stated that, with the increasing number of customers, the portion of bigger groups increases as more people with similar itineraries and time schedules can be grouped together.

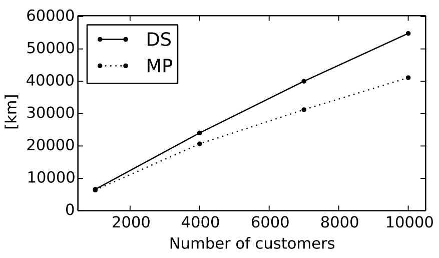
It can be seen that the majority of customers boards at most with three other passengers. Larger groups are more uncommon, and only the group with the maximum size is more frequent than the other large groups, as they cover cases where the cluster of customers with similar itineraries could have been even larger, e.g. commuters from a densely populated residential district to the city centre. Note that the group size can be higher than the actual maximum vehicle capacity, which is set to 9 to imitate minibuses. The actual assignment of passengers to minibuses is part of the vehicle routing phase, since only there the actual vehicle occupancy can be handled. As an example, a group size of 11 could be transported by three different vehicles: one with 5 spare seats, one with 3 spare seats and another one with at least 3 spare seats.

For our study area, Braunschweig city, 4000 passengers are already sufficient for a majority of people to share their rides. This also inherently reduces the number of necessary boarding and de-boarding service stops for the vehicles (Fig. 12b), since they have to stop only once for a group instead of stopping for every single customer. Naturally, the savings are higher when the demand is dense. With 5000 customers, the number of necessary stops is reduced by 33%, while it is halved at about 15,000 customers.

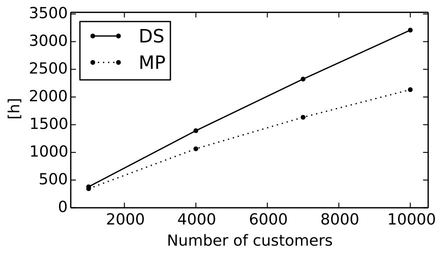
After the vehicle routing optimization phase, several further statistics about actual vehicle usage and trip times can be derived (cf. Table 2 in Appendix E). Figure 13 gives an impression about the potential benefits for the operator when using meeting points (i.e. the MP case) instead of offering pick-ups at the doorstep (i.e. the DS case). While the effect is relatively small in low demand cases, the benefits are more



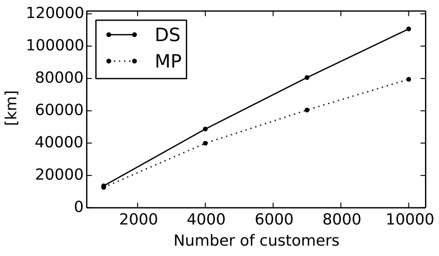
(a) Number of used vehicles



(b) Vehicle dead kilometers

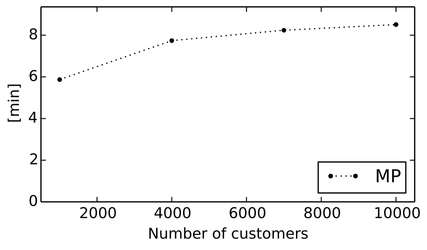


(c) Total vehicle hours

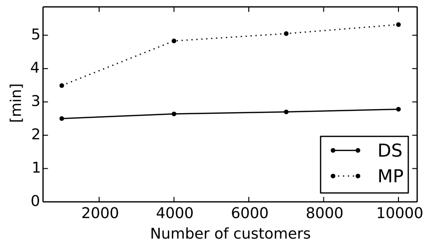


(b) Total vehicle kilometers

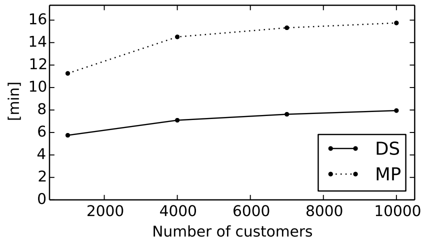
Fig. 13 Comparison between meeting point and doorstep simulation concerning vehicle statistics. *DS* Doorstep, *MP* meeting points



(a) Passenger average walking time



(b) Passenger average pickup wait time



(c) Passenger average detour time

Fig. 14 Comparison between meeting point and doorstep simulation concerning passenger statistics (*DS* doorstep, *MP* meeting points)

significant with increasing number of customer requests. All plots in Fig. 13 show a similar trend. In the 10,000 customers case, the savings in time, kilometre, and fleet size is up to 30%. In addition, there are fewer dead kilometres (the distance travelled when the vehicle is without a passenger).

Figure 14, on the other hand, shows the drawback for the passengers resulting from the usage of meeting points. Naturally, the walking time is an additional factor that has to be considered in the total travel times (Fig. 14a). The average walking times in our example range from 6 to 8 min, including the walking time from the alighting point to the destination. For the doorstep case, the walking time is obviously zero. In addition to the walking time, the average waiting times at the meeting points are higher compared to the doorstep case (Fig. 14b), since passengers likely have to wait for other fellow travellers. Here, the average waiting times for the pick-up are almost doubled when using meeting points, but the absolute values with about 3 to 5.5 min are comparably low. The total travel time differences between meeting and doorstep case can be seen in Fig. 14c.

6 Discussion

In general, the results confirm the benefits for operators of SDRT systems when using real-world meeting and divergence points for the pick-up and drop-off procedure. Fewer vehicles, less vehicle kilometers and a reduction of necessary boarding and alighting stops results in reduction of operational costs, especially for cases where the demand is high.

Using meeting points leads to improved operational efficiency for operators. In particular all measured metrics were improved in favour of the MP case compared to the DS case. In particular, in Fig. 13 we see similar improvement trends for all examined metrics: i.e. number of used vehicles, total vehicles operating hours, total traveled kilometers and total dead kilometers. A common trend can be observed as more customers use the SDRT-MP the benefits become more significant. This is due to the fact that the DS case shows linear growth in all costs, whereas the MP case leans towards logarithmic growth. This is expected since the MP set is fixed for a given area, which implies a fixed MP distribution, hence increasing the density of customers only increases customer aggregation at meeting points. Note that the number of customers cannot be increased indefinitely since in addition to SDRT-MP, users have the option to choose other modes of transport (private vehicle, public transport, etc.). Hence, the average trip size presented in Fig. 12a represents the possible ideal case. Interestingly, the simulation results also suggest a reduced computation time, which is another benefit to the operator. From Table 2, it can be seen that there are dramatical differences in running times between MP and DS cases during vehicle routing optimization, particularly for the cases with larger demand size.

Benefits for the operators should translate directly to lower price for the service which is directly beneficial to the users of SDRT-MP. Additionally, SDRT-MP offer a mobility service that is conceptually between public transport and a taxi service in terms of flexibility. Hence, this provides more options for the users.

On the other hand, the use of MP will likely result in increased total travel time. The main cause of this is the fact that the users have to walk to the MP and to wait for the other passengers with the same pick-up to arrive (Fig. 14a). This has to be balanced carefully during the meeting point selection to provide a good coverage of the service area with MP. It can also be seen in Figs. 14b and 14c that the average pickup time and the average detour time is worse for the MP users than it is for the DS users. The longer pickup time is caused by waiting for other passengers to arrive at the MP, since not all passengers arrive simultaneously. While the average walking time can only be improved with more MP, average pickup time and average detour time can be improved by improving optimization methods which is a topic for further research.

In literature, there are only very few applications using real-word meeting points. In the simulation of Häll et al. (2008) it was found that, in general, the DS case offers a better service for the customers, which can be confirmed by the results of this paper regarding travel time. However, they found no major differences in the results between the MP and the DS case and state that, according to their results, a door-to-door service can be offered without any noticeable loss in efficiency. This contradicts the findings of this paper, in which an improved operational efficiency has been demonstrated (Fig. 13). This discrepancy can be easily explained since meeting points in the study by Häll et al. (2008) are distributed on a rectangular grid that is spaced 75m apart, which arguably is too small for any differences to be seen.

Although the simulation experiment by Stiglic et al. (2015) does not include a routing phase, and focuses on ride-sharing, which allows only one boarding and alighting per vehicle, it is, nevertheless, interesting to draw a comparison. They conclude that the introduction of meeting points can improve a number of performance metrics, such as kilometer savings and an increase in the number of matched riders. In their simulations, the total walking time is, on average, between 8 and 9 min, which is comparable to the finding of this paper. Furthermore, they state that the average trip time for matched riders increases by approximately 12% due to the walking to or/and from a meeting point. In this paper, the travel time increase is up to 44%, which can be attributed to the relatively low total travel times (17.5 min for the DS case, 25.3 min for the MP case on average). Since Braunschweig is a small city and congestion is not modelled, all nodes of the city network can be reached within a short time, and hence meeting and waiting times have a high impact on the result.

The proposed solution is based on heuristics since it is not possible to compute the optimal solution for the tested instances. The heuristic solution can hence not claim to be optimal, and the gap to an optimal solution remains unknown. However, already the heuristic solution allows us to show the impact of a meeting point-based service compared to a door-to-door service. Both MP and DS cases are solved using the same methodology and we conjecture that the optimality gap should be equally present in both cases. Furthermore, as can be observed in Table 2 in Appendix E, the route optimization computation time demonstrated that the heuristics used to solve the vehicle routing problem do not scale well. Although this is not of importance for scientific research, we would like to emphasize that for production use more scalable vehicle routing solvers should be used. For that reason, the computational time of the approach and the absolute result quality is therefore not considered to be relevant.

Another issue is that the numerical experiments are focused on a single instance. The conclusions may therefore not be very robust to changes in the input data. For future research, it is hence desirable to expand the experiments to cover more cities.

In contrast to the algorithms of commercial SDRT services such as UberPOOL or Lyft Line (see Sect. 1 for more examples and references), the proposed algorithm groups the users in a batch system, offering routing results for a fixed point in time (static dial-a-ride problem). Since requests also arrive dynamically, it is essential to allow real-time matching results (dynamic dial-a-ride) for a commercial system. With the proposed algorithm, this can be achieved by frequent runs of the algorithm, if the computation time can be reduced. However, for a real-time analysis of meeting points new heuristics have to be developed that can deal with requests that arrive spontaneously and offer good meeting points. The concept of launch pads could offer a dynamic solution to the aforementioned problem (Rigby et al. 2016). Additionally, the commercial SDRT services seem to be solving a slightly different problem. Namely, it seems that their solutions have no optimization concept that involves multiple passengers and works by optimizing vehicle movement around the pickup locations of a single passenger and then assigning the passengers to vehicles. However, as the algorithms used in these services are not public, the authors can only infer their solutions based on their public announcements.

7 Conclusion

The objective of this paper was to evaluate if shared demand-responsive transportation (SDRT) would benefit from the introduction of meeting points (MP) for pickup and drop-off, when those points are methodologically extracted from map data by using GIS tools. The evaluation was performed by comparing the proposed method to the more traditional door-to-door based SDRT. It was found that major benefits for the operators can be expected if SDRT-MP is implemented.

This type of service fills the gap between mass transit systems and taxi systems. Such SDRT services are already available in the market (cf. UberPOOL, Lyft Line), meaning that integrated planning of transportation systems should also consider them. It can be concluded that the placement of MP greatly affects the user's satisfaction (indicated by metrics such as travel time, transport price, and additional walking time). Hence, the placement and the density of MP should be considered as an initial planning step and should be in focus for future transport planning methods.

The results of this paper have demonstrated that the use of real-world MP for SDRT service will offer a more efficient operation, unlike what was suggested in some of the literature. As a result, the users of SDRT-MP will receive benefit in the form of a reduced price for the service. In exchange, the users would be required to do extra walking and experience longer travel time overall. Arguably, transportation service that promotes active mode of transport should be encouraged as it offers a healthier lifestyle. In the case of passengers with mobility impairment, it is relatively straightforward to make service adjustments to accommodate their special needs, such as allowing door-to-door service and using wheelchair accessible vehicles. Several previous works have shown how multiple types of customers can be handled in a SDRT service, see Li et al. (2014); Kutadinata et al. (2019).

Several future research directions have been identified, including: sensitivity analysis of the methodology to different case studies; pricing mechanism that considers dynamic requests and meeting point allocation; integration with visual representation of possible meeting points (e.g. in Rigby et al. 2013); and social network-based clustering methodology (e.g. in Wang et al. 2019).

Acknowledgements This research has been supported by the German Research Foundation (DFG) through the Research Training Group Social Cars (GRK 1931), the Australian Research Council's Linkage Projects funding scheme (project number LP120200130), the Universities Australia and the German Academic Exchange Service (DAAD) under the Australia-Germany Joint Research Co-operation Scheme.

Appendix A: Preliminary note on notation and algorithms

Square brackets indicate access to a certain element in the vector or a set. For instance, $S[0]$ indicates an access to the first element of the set S. Additionally, note that the authors employ zero-based indexing which is common in C like programming languages. In many algorithms, we used mathematical notation for what are common algorithmic procedures. This allows algorithms to be more compact while still keeping the same level of information.

Appendix B: Clustering algorithm

Algorithm 1 Clustering algorithm

```
    Input
    $X$ – Set of all requests; $k^{\max}$ – Maximum cluster size
    Output
    $\mathcal{R}$ – Result set containing all clusters.
 1: procedure Clustering($X$, $k^{\max}$)
 2:     $\mathcal{R} \leftarrow \emptyset$
 3:     $\mathcal{C} \leftarrow \{\text{PopFront}(X)\}$ ▷ PopFront($X$) removes and returns the first element from $X$.
 4:     while $|X| > 0$ do
 5:         $x^* \leftarrow \arg\min_{x \in X} d(\mathcal{C}, x)$      ▷ Get the closest request. $d(\cdot,\cdot)$ is defined in Equation 3.
 6:         $\mathcal{C} \leftarrow \mathcal{C} \cup \{x^*\}$      ▷ Add to cluster.
 7:         $X \leftarrow X \setminus \{x^*\}$      ▷ Remove from available requests.
 8:         if $|\mathcal{C}| = k^{\max}$ then
 9:             $\mathcal{R} \leftarrow \mathcal{R} \cup \{\mathcal{C}\}$      ▷ Add cluster to the result set.
10:             $\mathcal{C} \leftarrow \{\text{PopFront}(X)\}$
11:         end if
12:     end while
13:     return $\mathcal{R}$
14: end procedure
```

Appendix B.1: Complexity analysis

Using \ for integer division and given:

- k – Maximum cluster size,
- n – Number of requests i.e. $n = |X|$,
- $N := n \backslash k$,

the upper bound for the number of computations can be obtained using the following equation:

$$\sum_{i=0}^{N} \alpha(i)(k - i \bmod k)(n - i \cdot k) \leq k \sum_{i=0}^{N} \alpha(i)(n - i \cdot k) \tag{9}$$

where:

$$\alpha(i) = \begin{cases} 0, & \text{if } i \backslash k = 0 \\ 1, & \text{otherwise.} \end{cases} \tag{10}$$

Notice that:

$$\sum_{i=0}^{N} (n - i \cdot k) \leq \frac{N}{2} \cdot (2n - N \cdot k) \tag{11}$$

Since (10) tells us that every k^{th} element of the sum is zero, i.e. the operation is of complexity $\mathcal{O}(1)$, we have:

$$\frac{N}{2} \cdot (2n - N \cdot k) \leq \frac{N - \frac{N}{k}}{2} \cdot (2n - N \cdot k) \tag{12}$$

Putting all this together in (9) and noticing that $N < \frac{n}{k}$, we get the complexity upper bound:

$$k \frac{N - \frac{N}{k}}{2} \cdot (2n - N \cdot k) \leq \frac{n^2 \cdot (k-1)}{2k} \leq \frac{n^2}{2} \tag{13}$$

Therefore, Algorithm 1 has time complexity of $\mathcal{O}(\frac{n^2}{2})$.

Appendix C: Meeting point candidates selection

Algorithm 2 describes the procedure for the meeting point determination for a given demand cluster. The mentioned *2-Combinations* function yields all possible paired combinations of a given set, e.g. *2-Combinations(a,b,c,d) = [a–bcd, b–acd, c–abd,* *d–abc, ab–cd, ac–bd, ad–bc]*. The algorithm returns the optimal combination in terms of number of subgroups and summed squared walking distances.

Algorithm 2 Meeting Point Determination

```
    Input
    $c$ – Passengers group set i.e. $c = \{\rho_1, \rho_2, \dots\}$.
    Output
    $\mathcal{S}^*$– Best combination of meeting and divergence points.
 1: procedure FindCombination($c$)                ▷ Procedure is called for every cluster.
 2:     $\mathcal{M} \leftarrow \mathcal{M}_{\rho_1} \cap \mathcal{M}_{\rho_2} \cap \dots$     ▷ Find common meeting points among the passengers.
 3:     $\mathcal{D} \leftarrow \mathcal{D}_{\rho_1} \cap \mathcal{D}_{\rho_2} \cap \dots$     ▷ Find common divergence points among the passengers.
 4:     $\mathcal{M}_{TF} \leftarrow \emptyset$, $\mathcal{D}_{TF} \leftarrow \emptyset$
 5:     $\mathcal{S}^* \leftarrow (|c|, \infty, \emptyset, \emptyset)$              ▷ Initialize current best solution.
 6:     if $|\mathcal{M}| \geq 1$ & $|\mathcal{D}| \geq 1$ then
 7:         // Check if at least one common MP exists.
 8:         for $\mu \in \mathcal{M}$ do
 9:             if $\max_{\rho \in c}(t^E_\rho(\mu)) \leq \min_{\rho \in c}(t^L_\rho(\mu))$ then
10:                 $\gamma(\mu) \leftarrow \sum_{\rho \in c} d^W(v^+_\rho \rightarrow \mu)^2$
11:                 $\mathcal{M}_{TF} \leftarrow \mathcal{M}_{TF} \cup (\gamma(\mu), \mu)$      ▷ Time feasible, add to set.
12:             end if
13:         end for
14:         // Check if at least one common DP exists.
15:         for $\delta \in \mathcal{D}$ do
16:             if $\max_{\rho \in c}(t^E_\rho(\delta)) \leq \min_{\rho \in c}(t^L_\rho(\delta))$ then
17:                 $\gamma(\delta) \leftarrow \sum_{\rho \in c} d^W(v^+_\rho \rightarrow \delta)^2$
18:                 $\mathcal{D}_{TF} \leftarrow \mathcal{D}_{TF} \cup (\gamma(\delta), \delta)$      ▷ Time feasible, add to set.
19:             end if
20:         end for
21:         // Compute time feasibility for divergence points $\mathcal{D}_{TF}$ same as $\mathcal{M}_{TF}$.
22:         if $|\mathcal{M}_{TF}| \geq 1$ & $|\mathcal{D}_{TF}| \geq 1$ then
23:             // Check if time feasible common MP and DP exists.
24:             $\gamma^*(\mu), \mu^* \leftarrow \min(\mathcal{M}_{TF})$          ▷ Find $\mu$ with minimal cost.
25:             $\gamma^*(\delta), \delta^* \leftarrow \min(\mathcal{D}_{TF})$          ▷ Find $\delta$ with minimal cost.
26:             $\mathcal{S}^* \leftarrow (1, \gamma^*(\mu) + \gamma^*(\delta), \mu^*, \delta^*)$   ▷ Return combined cost, $\mu$ and $\delta$.
27:         end if
28:     else                                ▷ No common meeting and divergence point exists.
29:         for $c1, c2 \in$ 2-Combinations($c$) do
30:             $\mathcal{S}_1 \leftarrow$ FindCombination(c1)
31:             $\mathcal{S}_2 \leftarrow$ FindCombination(c2)
32:             if $\mathcal{S}_1 \cup \mathcal{S}_2$ better than $\mathcal{S}^*$ then    ▷ Better = Less separate groups.
33:                 $\mathcal{S}^* \leftarrow \mathcal{S}_1 \cup \mathcal{S}_2$
34:             end if
35:         end for
36:     end if
37:     return $\mathcal{S}^*$
38: end procedure
```

Algorithm 3 Recursive Alternative Meeting Point Search

```
    Input
    $\mathcal{M}$ – Set of meeting points that all passengers of a trip can reach.
    $\mathcal{M}_C$ – Set of meeting points determined by Algorithm 2 such that $\mathcal{M}_C \subset \mathcal{M}$.
    $A, \sigma$ – Travel time ratio matrix and threshold, respectively.
    Output
    $\mathcal{M}_C$ – Set of meeting points to be considered.
 1: procedure FindMPAlternatives($\mathcal{M}, \mathcal{M}_C, A, \sigma$)
 2:     $\mathcal{R} \leftarrow \emptyset$
 3:     for $\alpha \in \mathcal{M} \setminus \mathcal{M}_C$ do
 4:         $\mathcal{S} \leftarrow \{A[\alpha][\beta] \mid \forall \beta \in \mathcal{M}_C\}$
 5:         $\mathcal{R} \leftarrow \mathcal{R} \cup (\min(\mathcal{S}), \alpha)$        ▷ Minimum among all already considered points.
 6:     end for
 7:     $\gamma \leftarrow \max(\mathcal{R})$        ▷ Point with biggest minimal value.
 8:     if $\gamma[0] \geq \sigma$ then        ▷ Check if threshold is satisfied.
 9:         $\mathcal{M}_C \leftarrow \mathcal{M}_C \cup \gamma[1]$        ▷ Add to the set of considered points.
10:         return FindMPAlternatives($\mathcal{M}, \mathcal{M}_C, A, \sigma$)
11:     end if
12:     return $\mathcal{M}_C$        ▷ Set of meeting points to be considered.
13: end procedure
```

Appendix D: Vehicle routing optimization

Appendix D.1: Route optimization

Let the index set $\mathcal{V} = \{0, \mathcal{S}, \mathcal{M}, \mathcal{D}\}$ be assigned to these nodes, where $\mathcal{S}$ is the set of v starting nodes of vehicles, $\mathcal{M}$ is the set of n pick-up vertices, $\mathcal{D}$ is the set of corresponding n drop-off vertices. Node 0 is the depot, and a demand is represented by a pair of pick-up and drop-off points $(i, n+i)$. These nodes are connected with directed edges, where each edge indicates a possible route to be traversed by the vehicles. Let $\mathcal{E}$ be defined as the set of all directed edges in the network. A directed edge $\epsilon_{i,j} \in \mathcal{E}$ connects a pair of nodes, from Node i to Node j, where i, j have the following relation: $j \in \{\{0\} \cup \mathcal{M} \mid \forall i \in \mathcal{S}\}$, $j \in \{\mathcal{M} \cup \mathcal{D} \mid \forall i \in \mathcal{M}\}$, $j \in \{\mathcal{V} \setminus (\mathcal{S} \cup \{i-n\}) \mid \forall i \in \mathcal{D}\}$.

Each node has several parameters that define the trip demand. Similarly, each directed edge has its own associated distance and travel time parameters. To take into account the multiple alternative meeting and divergence points, some of these parameters are defined as functions of the chosen meeting/divergence points. For each $i \in \mathcal{M} \cup \mathcal{D}$, let $N_i \in \mathbb{N}$ be the number of alternative meeting/divergence points for node i. Note that these alternative meeting points are not modelled as nodes and are not part of the graph; rather they can be seen as parameters for each node in the sets $\mathcal{M}$ and $\mathcal{D}$. Also, note that some of these alternative points may, in real life, correspond to the same location, in which case the travel time and distance between them is zero. Next, define a vector of integer variables $m = \begin{bmatrix} m_1 & m_2 & \dots & m_{v+2n+1} \end{bmatrix}$, where $m_i \in \{1, 2, \dots, N_i\}$ is the chosen meeting/divergence point alternative for node i. Thus, m is part of the optimization decision variables specifying the meeting point selection.

Having established this, all the graph parameters can be defined. Let $[t_i^E(m_i), t_i^L(m_i)]$ be the associated time-window of node i when m_i is chosen. Note that the time windows are treated as soft constraints. Furthermore, let $d_{i,j}(m_i, m_j)$ and

$t_{i,j}(m_i, m_j)$ be the travel distance and time from node i to node j when using m_i and m_j. Observe that, as mentioned in Sect. 3.5, the meeting point selection influences the route optimization as the time windows, distances, and travel times are defined as functions of the meeting point decision variable, m_i. In addition, each node has an associated service time s_i and a load q_i. For $i \in \{0\} \cup \mathcal{S}$ (the depot and the starting points), $s_i = q_i = 0$, $N_i = 1$, and $[t_i^E(1), t_i^L(1)] = [0, \infty]$. For the other nodes, $s_i = t_*^{\text{serv}}$. Moreover, in order to keep track of the vehicle loads, let q_i^k be the passenger load on-board vehicle k when it leaves node i.

Finally, the optimization formulation is ready to be presented and the decision variables can be defined. Let u_i^k denote the time vehicle k starts servicing node i. The starting point of each vehicle k is similarly denoted with the index k and $s_k = 0$. Hence, the variable u_k^k (which technically is the start of the service time of vehicle k at node k) indicates the first departure time of vehicle k from its starting point. Similarly, u_0^k represents the final arrival time at the depot. The binary variable $x_{i,j}^k$ is defined to decide whether vehicle k traverses from node i to node j. Thus, the decision variables are the service time u_i^k, the binary variable $x_{i,j}^k$, and the meeting points m_i. The optimization problem is formulated as follows:

$$\begin{aligned}\min_{x,u,m} \sum_{i \in \mathcal{M} \cup \mathcal{D}} & \left[c_*^{\text{wait}} \max\{u_i^k - t_i^E(m_i), 0\}^\alpha + c_*^{\text{late}} \max\{u_i^k - t_i^L(m_i), 0\}^\beta\right] \\ & + \sum_{k \in \mathcal{S}} \left[c_*^{\text{vehi}} \operatorname{sgn}\left(u_0^k\right) + \sum_{i,j \in \mathcal{V}} c_*^{\text{dist}} x_{i,j}^k d_{i,j}(m_i, m_j)\right]\end{aligned} \tag{14}$$

subject to:

$$x_{i,j}^k \in \{0, 1\} \quad \forall i,j \in \mathcal{V}, k \in \mathcal{S}, \tag{15}$$

$$u_i^k \in \mathbb{R}_{\geq 0} \quad \forall i \in \mathcal{V}, k \in \mathcal{S}, \tag{16}$$

$$m_i \in \mathbb{N}^+_{\leq N_i} \quad \forall i \in \mathcal{V}, \tag{17}$$

$$\sum_{k \in \mathcal{S}} \sum_{j \in \mathcal{V}} x_{i,j}^k = 1 \quad \forall i \in \mathcal{M}, \tag{18}$$

$$\sum_{j \in \mathcal{V}} x_{i,j}^k = \sum_{j \in \mathcal{V}} x_{n+i,j}^k \quad \forall i \in \mathcal{M}, \forall k \in \mathcal{S}, \tag{19}$$

$$\sum_{i \in \mathcal{V}} x_{k,i}^k = \sum_{i \in \mathcal{V}} x_{i,0}^k = 1 \quad \forall k \in \mathcal{S}, \tag{20}$$

$$\sum_{j \in \mathcal{V}} x_{i,j}^k = \sum_{j \in \mathcal{V}} x_{j,i}^k \quad \forall i \in \mathcal{M} \cup \mathcal{D}, \forall k \in \mathcal{S}, \tag{21}$$

$$u_j^k \geq x_{i,j}^k \cdot \max\left\{u_i^k + s_i + t_{i,j}(m_i, m_j), t_j^E(m_j)\right\} \quad \forall i,j \in \mathcal{V}, \forall k \in \mathcal{S}, \tag{22}$$

$$q_j^k = x_{i,j}^k \cdot (q_i^k + q_j) \quad \forall i,j \in \mathcal{V}, \forall k \in \mathcal{S}, \tag{23}$$

$$u_{i+n}^k \geq u_i^k \quad \forall i \in \mathcal{M}, \forall k \in \mathcal{S}, \tag{24}$$

$$0 \leq q_i^k \leq q_* \quad \forall i \in \mathcal{V}, \forall k \in \mathcal{S}. \tag{25}$$

where sgn(·) is a function that returns the sign of the input scalar. For a further parameter description we also refer to Table 1.

The formulation ensures that each customer is picked up only once and is dropped off at the destination by imposing (18) and (19). Furthermore, a vehicle has to start at its corresponding starting point and ends its route at the depot as enforced by (20) and (21). Constraint (22) provides a lower bound on the arrival time of a vehicle at a node and (24) ensures that the drop-off occurs after the corresponding pick-up. Note that the lower bound (22) allows u_i^k to be zero if vehicle k never visits node i. Finally, (23)–(25) are constraints for the passenger load of each vehicle.

The first summation in the objective function is the "service level cost". The first term penalises the "wait time", which is defined as the difference between the arrival time and the start of a time window, and the second term penalises the "late time", defined as the difference between the arrival time and the end of a time window. The parameters α and β are introduced to enable various polynomial forms of penalty terms. For instance, a quadratic term ($\alpha = 2$) can be used to penalize longer wait/late times more than short wait/late times. With this formulation, the time windows are treated as penalty terms instead of hard constraints, which is different to typical formulations in the literature (Bent and van Hentenryck 2004; Cordeau and Laporte 2007; Ropke and Cordeau 2009; Baldacci et al. 2011).

The first term in the second summation is the capital cost of each vehicle used. Recall that the formulation enforces that all vehicles start at their own starting points and end at the depot due to (20) and (21). However, if a vehicle k is actually unused, its starting point (Node k) coincides with the depot location (Node 0) and it will not incur a capital cost according to (22), in a sense that it allows $u_0^k \geq 0$. Thus, the optimization would naturally choose $u_0^k = 0$. Finally, a route length minimization is taken into account by the last term of the objective function.

As mentioned in Section 3.5, the optimization problem (14)–(25) is solved heuristically with Algorithms 4 and 5.

Algorithm 4 Trip allocation neighbourhood search

```
    Input
    $\Theta^k := \{\tau_i\}$ ▷ where $\tau_i$ represents trip $i$ and $\Theta^k$ is the set of all trips assigned to vehicle $k$
    $\mathcal{R}^k$ ▷ where $\mathcal{R}^k$ is the route of vehicle $k$
    Output
    $\Theta^k_{\text{best}}$
    $\mathcal{R}^k_{\text{best}}$
 1: procedure TripAllocationNeighbourhoodSearch($\Theta^k$, $\mathcal{R}^k$)
 2:     $\Theta^k_{\text{best}} \leftarrow \Theta^k, \forall k.$
 3:     $\mathcal{R}^k_{\text{best}} \leftarrow \mathcal{R}^k, \forall k.$
 4:     for a pre-specified number of iterations do
 5:         $\Theta^{\text{neighbour},k}_{\text{best}} \leftarrow \Theta^k, \forall k.$
 6:         $\mathcal{R}^{\text{neighbour},k}_{\text{best}} \leftarrow \mathcal{R}^k, \forall k.$
 7:         for $\gamma \in \{1, 2, \dots, \text{neighbourhood size}\}$ do
 8:             Randomly choose a trip $\tau_i \in \cup_k \Theta^k$.
 9:             $k \leftarrow \eta_1$ ▷ $\eta_1$ is the vehicle index of the chosen trip.
10:             $\beta \leftarrow \{k \mid \Theta^k \neq \emptyset\} \setminus \{\eta_1\} \cup \min\{k \mid \Theta^k \neq \emptyset\}.$
11:             Reinsert the trip to a random vehicle $k \in \beta$. ▷ greedy insertion
12:             $k \leftarrow \eta_2$ ▷ $\eta_2$ is the vehicle index being inserted.
13:             $\Theta^{\text{neighbour},k}_{\gamma} \leftarrow$ the new trip allocation after reinsertion, $\forall k$
14:             $\mathcal{R}^{\text{neighbour},k}_{\gamma} \leftarrow \mathcal{R}^k, \forall k \notin \{\eta_1, \eta_2\}$
15:             Re-optimise $\mathcal{R}^{\eta_1}$ and $\mathcal{R}^{\eta_2}$ using Algorithm 5.
16:             $\mathcal{R}^{\text{neighbour},\eta_1}_{\gamma} \leftarrow \mathcal{R}^{\eta_1}$
17:             $\mathcal{R}^{\text{neighbour},\eta_2}_{\gamma} \leftarrow \mathcal{R}^{\eta_2}$
18:             if $\sum_k J(\mathcal{R}^{\text{neighbour},k}_{\gamma}) < \sum_k J(\mathcal{R}^{\text{neighbour},k}_{\text{best}})$ then
19:                 $\Theta^{\text{neighbour},k}_{\text{best}} \leftarrow \Theta^{\text{neighbour},k}_{\gamma}, \forall k.$
20:                 $\mathcal{R}^{\text{neighbour},k}_{\text{best}} \leftarrow \mathcal{R}^{\text{neighbour},k}_{\gamma}, \forall k.$
21:             end if
22:         end for
23:         if $\sum_k J(\mathcal{R}^{\text{neighbour},k}_{\text{best}}) < \sum_k J(\mathcal{R}^k_{\text{best}})$ then
24:             $\Theta^k_{\text{best}} \leftarrow \Theta^{\text{neighbour},k}_{\text{best}}, \forall k.$
25:             $\mathcal{R}^k_{\text{best}} \leftarrow \mathcal{R}^{\text{neighbour},k}_{\text{best}}, \forall k.$
26:         end if
27:         $\Theta^k \leftarrow \Theta^{\text{neighbour},k}_{\text{best}}, \forall k.$
28:         $\mathcal{R}^k \leftarrow \mathcal{R}^{\text{neighbour},k}_{\text{best}}, \forall k.$
29:     end for
30:     return $\Theta^k_{\text{best}}$, $\mathcal{R}^k_{\text{best}}$
31: end procedure
```

Appendix D.2: Route concatenations

Let R be the set of all routes produced by the route optimization process. A route $r \in R$ is defined by the tuple $\{t^s, b^s, t^e, b^e\}$, where t^s is the start of the service time of the first stop of r (that is not the vehicle starting point), b^s is the location of the first stop of r, t^e is the end of the service time of the last stop of r (that is not the depot), and b^e is the location of the last stop of r. Moreover, $t^{\text{driv}}(b^e_i \rightarrow b^s_j)$ is the vehicle travel time from the last stop of r_i to the first stop of r_j. Furthermore, define the decision variable $x_{i,j}$, which is equal to one if r_j is appended to the end of r_i, and zero otherwise. The optimization aims at maximizing the number of concatenation as follows:

$$\max_{x} \quad \sum_{i,j} x_{i,j} \tag{26}$$

subject to:

$$t_i^e + t^{\text{driv}}(b_i^e \to b_j^s) \leq t_j^s, \qquad \forall i, j. \tag{27}$$

The constraint is applied to ensure that the resulting route allows enough time for the vehicle to reach the first stop of the subsequent route from the last stop of the preceding one.

Algorithm 5 Route neighbourhood search for a single vehicle

```
    Input
    $\mathcal{R}$ – Initial route set defined as $\mathcal{R} := \{s_1, s_2, \dots, s_N\}$ where $s_i$ is the $i^{\text{th}}$ stop of a vehicle.
    Output
    $\mathcal{R}^*$ – Optimized route set.
 1: procedure RouteNeighbourhoodSearch($\mathcal{R}$)
 2:     $\mathcal{R}^* \leftarrow \mathcal{R}$
 3:     for a pre-specified number of iterations do
 4:         for $\gamma \in \{1, 2, \dots, \text{neighbourhood size}\}$ do
 5:             $s \sim \mathcal{U}(\mathcal{R})$                    ▷ Pick stop $s$ with uniform probability.
 6:             $s' \sim \mathcal{U}(\{n \mid n \in \text{neighbouring stops of } s\})$
 7:             $\mathcal{R}^\gamma \leftarrow$ RemoveAndReplaceStop($s \to s'$)
 8:             $\mathcal{R} = \underset{r \in \{\mathcal{R}^*, \mathcal{R}^\gamma\}}{\arg\min} \; J(r)$          ▷ $J(\cdot)$ is a single vehicle objective function.
 9:         end for
10:     end for
11:     return $\mathcal{R}^*$
12: end procedure
```

Appendix E: Results

See Table 2 here.

Table 2 Results for the MP *meeting point* and DS *doorstep* case

No. of riders	1000		4000		7000		10000	
	MP	DS	MP	DS	MP	DS	MP	DS
Objective function value [–]	498,835	413,294	2,319,368	1,632,910	4,086,774	2,701,750	3,815,542	3,895,434
Total vehicles' travel distance [km]	12,594	13,536	39,915	48,723	60,485	80,570	79,470	110,655
Total vehicles' operating time [h]	344	379	1065	1392	1634	2324	2133	3208
No. of vehicles used [–]	184	198	602	794	931	1296	1248	1871
Total dead travel distance [km]	6,357	6,603	20,668	24,032	31,222	40,016	41,118	54,806
Total idle time [h]	36	32	91	91	134	136	161	160
Total empty-idle time [h]	33	28	84	71	121	106	145	119
Pick-up waiting time [min/trip]	2.05	–	1.73	–	1.54	–	1.39	–
Pick-up waiting time [min/passenger]	3.49	2.50	4.83	2.64	5.05	2.70	5.32	2.77
Detour time [min/trip]	11.26	–	14.51	–	15.32	–	15.75	–
Detour time [min/passenger]	2.17	5.75	2.42	7.09	2.56	7.62	2.4	7.95
Walk time [min/passenger]	5.87	–	7.74	–	8.24	–	8.51	–
Route optimization computation time [h]	0.94	3.15	3.12	9.79	4.51	16.50	5.30	25.80

References

Aissat K, Oulamara A (2014) Dynamic ridesharing with intermediate locations. In: Computational intelligence in vehicles and transportation systems (CIVTS), 2014 IEEE Symposium on, IEEE, pp 36–42

Aissat K, Oulamara A (2015) Meeting locations in real-time ridesharing problem: a buckets approach. In: Operations research and enterprise systems, Springer, pp 71–92

Aïvodji UM, Gambs S, Huguet MJ, Killijian MO (2016) Meeting points in ridesharing: a privacy-preserving approach. Transp Res Part C Emerg Technol 72:239–253

Balardino AF, Santos AG (2016) Heuristic and exact approach for the close enough ridematching problem. In: Hybrid intelligent systems, Springer International Publishing, pp 281–293. https://doi.org/10.1007/978-3-319-27221-4_24

Baldacci R, Bartolini F, Mingozzi A (2011) An exact algorithm for the pickup and delivery problem with time windows. Oper Res 59(2):414–426

Bent R, van Hentenryck P (2004) A two-stage hybrid local search for the vehicle routing problem with time windows. Transp Sci 38(4):515–530

Chen W, Mes M, Schutten J, Quint J (2016) A ride-sharing problem with meeting points and return restrictions. BETA Working Paper Series 516. http://doc.utwente.nl/101813/1/wp_516.pdf

Cordeau JF, Laporte G (2007) The dial-a-ride problem: models and algorithms. Ann Oper Res 153(1):29–46. https://doi.org/10.1007/s10479-007-0170-8

Correa O, Ramamohanarao K, Tanin E, Kulik L (2017) From ride-sourcing to ride-sharing through hotspots. In: Proceedings of the 2017 MobiQuitous, ACM. https://doi.org/10.1145/3144457.3144483

Czioska P, Trifunović A, Dennisen S, Sester M (2017) Location- and time-dependent meeting point recommendations for shared interurban rides. J Location Based Serv 11(3–4):181–203. https://doi.org/10.1080/17489725.2017.1421779

Eser E, Monteil J, Simonetto A (2018) On the tracking of dynamical optimal meeting points. In: Proceedings of the 15th IFAC symposium on control in transportation systems

Giles-Corti B, Vernez-Moudon A, Reis R, Turrell G, Dannenberg AL, Badland H, Foster S, Lowe M, Sallis JF, Stevenson M et al (2016) City planning and population health: a global challenge. The Lancet 388(10062):2912–2924. https://doi.org/10.1016/S0140-6736(16)30066-6

Goel P, Kulik L, Ramamohanarao K (2016) Privacy-aware dynamic ride sharing. ACM Trans Spatial Algorithms Syst 2(1):4:1–4:41. https://doi.org/10.1145/2845080

Goel P, Kulik L, Ramamohanarao K (2017) Optimal pick up point selection for effective ride sharing. IEEE Trans Big Data 3(2):154–168. https://doi.org/10.1109/TBDATA.2016.2599936

Griswold A (2017) Why it matters that Uber and Lyft are becoming more like public transit. https://web.archive.org/web/20180309071444/https://qz.com/1022789/why-it-matters-that-uber-and-lyft-are-becoming-more-like-public-transit/. Accessed 11 Apr 2018

Häll CH, Lundgren JT, Värbrand P (2008) Evaluation of an integrated public transport system: a simulation approach. Arch Transp 20(1–2):29–46

Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning, 2nd edn. Springer, New York

Hess DB (2012) Walking to the bus: perceived versus actual walking distance to bus stops for older adults. Transportation 39(2):247–266

Kutadinata R, Thompson R, Winter S (2019) Passenger-freight demand responsive transport services: A dynamic optimisation approach. In: Proceedings of the 26th intelligent transport systems World Congress, to be presented

Li B, Krushinsky D, Reijers H, van Woensel T (2014) The share-a-ride problem: people and parcels sharing taxis. Eur J Oper Res 238(1):31–40

Mageean J, Nelson JD (2003) The evaluation of demand responsive transport services in Europe. J Transp Geogr 11(4):255–270. https://doi.org/10.1016/S0966-6923(03)00026-7

Martínez LM, Viegas JM, Eiró T (2014) Formulating a new express minibus service design problem as a clustering problem. Transp Sci 49(1):85–98

Millward H, Spinney J, Scott D (2013) Active-transport walking behavior: destinations, durations, distances. J Transp Geogr 28:101–110

Municipality of Braunschweig (2016) Open geodata. License: dl-de/by-2-0. http://www.braunschweig.de/opengeodata

Najmi A, Rey D, Rashidi TH (2017) Novel dynamic formulations for real-time ride-sharing systems. Transp Res Part E: Logist Transp Rev 108:122–140. https://doi.org/10.1016/j.tre.2017.10.009

Navidi Z, Ronald N, Winter S (2016) Comparing demand responsive and conventional public transport in a low demand context. In: 2016 IEEE international conference on pervasive computing and communication workshops (PerCom Workshops), pp 1–6. https://doi.org/10.1109/PERCOMW.2016.7457089

Nelson JD, Wright S, Masson B, Ambrosino G, Naniopoulos A (2010) Recent developments in flexible transport services. Res Transp Econ 29(1):243–248

Parragh SN, Doerner KF, Hartl RF (2008) A survey on pickup and delivery problems. J für Betriebswirtschaft 58(1):21–51

Rigby M, Winter S (2016) Usability of an opportunistic interface concept for ad hoc ride-sharing. Int J Cartogr 2(2):115–147. https://doi.org/10.1080/23729333.2016.1145040

Rigby M, Krüger A, Winter S (2013) An opportunistic client user interface to support centralized ride share planning. In: Proceedings of the 21st ACM SIGSPATIAL international conference on advances in geographic information systems, ACM, pp 34–43

Rigby M, Winter S, Krüger A (2016) A continuous representation of ad hoc ridesharing potential. IEEE Trans Intell Transp Syst 17(10):2832–2842. https://doi.org/10.1109/TITS.2016.2527052

Ropke S, Cordeau JF (2009) Branch and cut and price for the pickup and delivery problem with time windows. Transp Sci 43(3):267–286

Ropke S, Pisinger D (2006) An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transp Sci 40(4):455–472

Stiglic M, Agatz N, Savelsbergh M, Gradisar M (2015) The benefits of meeting points in ride-sharing systems. Transp Res Part B Methodol 82:36–53. https://doi.org/10.1016/j.trb.2015.07.025

Wang Y, Kutadinata R, Winter S (2019) The evolutionary interaction between taxi-sharing behaviours and social networks. Transp Res Part A Policy Pract 119:170–180