# The Line-Based Dial-a-Ride Problem

## Kendra Reiter[1] ✉ 🄳

Department of Computer Science, University of Würzburg, Germany

## Marie Schmidt ✉ 🄳

Department of Computer Science, University of Würzburg, Germany

## Michael Stiglmayr ✉ 🄳

Department of Mathematics and Informatics, University of Wuppertal, Germany

──── **Abstract** ────

On-demand ridepooling systems offer flexible services pooling multiple passengers into one vehicle, complementing traditional bus services. We propose a transportation system combining the spatial aspects of a fixed sequence of bus stops with the temporal flexibility of ridepooling. In the *line-based Dial-a-Ride problem (liDARP)*, vehicles adhere to a fixed, ordered sequence of stops in their routes, with the possibility of taking shortcuts and turning if they are empty. We propose three MILP formulations for the liDARP with a multi-objective function balancing environmental aspects with customer satisfaction, comparing them on a real-world bus line. Our experiments show that the formulation based on an Event-Based graph is the fastest, solving instances with up to 50 requests in under one second. Compared to the classical DARP, the liDARP is computationally faster, with minimal increases in total distance driven and average ride times.

## 1 Introduction

Line-based bus services are able to pool a large number of transportation requests along popular trajectories, thus contributing towards reducing mobility-related emissions when people decide to take the bus instead of the car. However, when there is little demand (e.g., in rural areas or during off-peak periods), buses often run infrequently and almost empty, so that the described benefits do not materialize. Ridepooling approaches, where multiple passengers are pooled into a shared vehicle, accepting a slight detour compared to their direct route, are often proposed to complement line-based bus services for these scenarios. These approaches are inefficient where they do not succeed to pool requests sufficiently.

We formalize a conceptual approach called *line-based ridepooling* which combines the spatial aspect of a classical line-based bus service with the temporal flexibility of on-demand ridepooling, inspired by real-life examples, including the FLEX'HOP 72[2] in France, and the NAHBUS[3] and the Rufbus[4] in Germany. In line-based ridepooling, we consider a fixed and ordered sequence of bus stops (defined, e.g., by a prior operating regular bus line) which we use as pick-up and drop-off locations. In contrast to a classical line-based bus, our vehicles have the flexibility to skip stops and take shortcuts, enabling tailored pick-up and drop-off

---

[1] corresponding author

[2] `https://www.cts-strasbourg.eu/fr/se-deplacer/transport-a-la-demande/`

[3] `https://www.nahbus.de/rufbus`

[4] `https://rufbus.nordfriesland.de/Rufbus-Nördliches-u-südliches-Nordfriesland/`

times dependent on the specific customer requests. However, in contrast to ridepooling, we use the spatial structure of the given line as a sort of service promise, ensuring that passengers are only transported towards their destination along the line. In particular, vehicles may not turn with passengers on board. We call this the *directionality property*. We aim to achieve a transportation mode which is more efficient and provides a higher quality of service than the classical line-based bus, especially in areas with low demand or during off-peak times.

In this paper, we define a new optimization problem to serve passenger requests in line-based ridepooling. Due to its similarity to the general Dial-a-Ride problem (DARP), from which it differs by the directionality property and the underlying geography, we call this problem the *line-based Dial-a-Ride problem (liDARP)*. Here, we study the static variant, where all requests are known ahead-of-time, and consider a homogeneous fleet of vehicles.

We observe that, due to the directionality property, each vehicle route can be decomposed into a number of sublines, separated by vehicle turns, with each transported passenger assigned to exactly one subline. We introduce and compare three mixed-integer linear programming (MILP) formulations for the liDARP that exploit this property. The first formulation, presented in Section 4.1, explicitly models sublines and the assignment of passengers to them. The second and third formulation, presented in Section 4.2 and Section 4.3, are based on Cordeau's classic 3-index Location-Based formulation [7] and the Event-Based model introduced by Gaul et al. in [14]. Section 5 discusses computational results.

Our contribution is threefold: First, we present a general problem definition for the liDARP, an approach to organizing passenger transport with the potential to combine benefits from a line-based public transport and on-demand transportation. Second, we develop and present three MILP formulations for the liDARP. Third, we compare these three models on synthetic test instances.

## 2    Related Work

Traditional modes for passenger transportation like the bus, metro, or train, operate based on lines (prescribing the sequence of stops visited) and timetables (prescribing the timing of each stop) or frequencies (prescribing the distance to be kept between individual vehicles on a line). While public transport planning often takes a network perspective, there are also many contributions that study timetabling or frequency setting on an individual line with the objective to find an optimal balance between service quality and operator cost, see, e.g., [18, 19] and the references therein. Gkiotsalitis et al. [15] present a model that allows to establish regularly operating sublines within a longer line to deal with inhomogenous demand along the line. Aktaş et al. [1] study a situation where selected stops are assigned to an *express service*, forming a shorter and quicker route. Their goal is to determine which vehicles should perform this express service during morning rush hour, based on expected demand. While still a rather uncommon strategy during the *planning* of public transport operations, short-turning and stop-skipping are common *control* strategies in transit systems to mitigate effects like vehicle bunching and overcrowding, see [18].

The literature on Dial-a-Ride problems (synonymously called *ridepooling*, *on-demand* bus services, or *demand-responsive transport*) is extensive, with in-depth overviews of the current state being provided by Cordeau and Laporte [9] (until 2007) and Ho et al. [17] (2007 until 2018). Typography and variants are discussed in Molenbruch et al. [20], whence this paper is concerned with the static, homogeneous, multi-objective approach, compromising the conflicting goals of system efficiency (including environmental aspects) and user experience.

Early approaches towards exact solution methods to the DARP were carried out by

Psaraftis in [25] and [26]. Cordeau [7] proposes a 3-index arc-based mixed-binary linear program for the standard DARP, which was adapted to a 2-index formulation by Røpke et al. [29]. Røpke et al. propose a branch-and-cut approach which is tested on a large number of benchmark instances. Parragh [22] constructs further valid inequalities related to capacity restrictions, integrating these into a branch-and-cut framework as well as a variable neighborhood search heuristic, based on both the 3-index and 2-index formulations. Gschwindt and Irnich [16] develop an exact branch-and-cut-and-price approach, which solves all instances of the benchmark set introduced by [29] exactly. Recently, Rist and Forbes [28] propose a branch-and-cut framework where a DARP route is broken in multiple *fragments*, which are paths between a request's pick-up and drop-off where the vehicle has a non-empty load. Then, a route is created as a combination of fragments. Gaul et al. [14] propose a new MILP formulation, relying on an *Event-Based* graph with nodes representing pick-up/drop-off events denoting a feasible user allocation of the corresponding vehicle and edges connecting feasible transitions between events.

Next to exact methods, many papers consider heuristic solution methods to solve the DARP, including metaheuristics such as simulated annealing [4, 27], adaptive large neighborhood search [23, 24, 30], and tabu search [3, 8].

We are aware of only three publications where the DARP is studied in combination with an underlying line structure: Archetti et al. [2] restate and prove results from the dissertation of Busch [6], showing that the Vehicle Routing Problem on the line is NP-hard, both with an unlimited and a limited fleet of fixed capacity. A complexity classification of DARP variants has been proposed by de Paepe et al. [10], establishing a scheme akin to scheduling problems. They examine variants on the line geography, showing that the DARP on the line with one vehicle of capacity one is solvable in polynomial time. The DARP on a line with multiple homogeneous vehicles of fixed capacity $\geq 1$ is NP-complete, which has been shown by Bjelde et al. [5] based on a reduction from the Circular Arc Coloring problem. All three papers are focused on exploring the complexity of the problem, where they consider only the special case with equally spaced stations and do not allow for shortcuts.

## 3    Problem Description

We consider a set of $\kappa$ vehicles of capacity $Q_{\max}$ that operate on a bus line, specified by a sequence of bus stops $H = (1, \ldots, n)$, to transport $m$ stop-to-stop passenger requests $R$.

The vehicles do not need to traverse the whole line in each route, but are allowed to take short-cuts (including skipping stops at which no passenger wants to board or alight), to wait, and to turn at any stop, the latter of which may not be done with passengers on board. In this way, we guarantee that the *directionality property* is fulfilled, i.e., each passenger, at all times, is transported towards their direction with respect to the sequence of stops defined by the bus line. Pairwise (time) distances $t_{i,j}$ between all stops $i, j \in H$ are given, with $t_{i,i} := t_{\text{turn}}$ denoting the turn time at $i \in H$. These distances respect the triangle inequality.

Each request $r \in R$ specifies an origin stop $o_r \in H$, a destination stop $d_r \in H$, a time window $[e_r, \ l_r]$, a load (number of passengers in the request) $q_r$, and a service time $b_r$ for boarding and alighting. We assume that boarding is synchronous, i.e., if one request's destination is another request's origin, and both requests are transported by the same vehicle, we require that the first request alights before the second boards. This reflects the widely accepted standard boarding procedure on public transit systems. Furthermore, passengers do not transfer between vehicles.

We make two *service promises* to our accepted passengers regarding 1) their total travel

time and 2) their waiting time that have to be respected. For the former, we guarantee that the passenger's total ride time $L_r$ will not exceed the time needed to travel the direct route (between their origin and destination) by a pre-specified factor, the *excess factor* $\alpha$, i.e., $L_r^{\max} := \alpha \cdot t_{o_r,d_r}$. For the latter, we ensure that the actual pick-up (resp. drop-off) time is not more than $\beta$ minutes later (resp. earlier) than the specified earliest pick-up (resp. latest drop-off) time.

Our objective is to create a reliable service for customers and, at the same time, integrate environmental aspects by reducing emissions compared to passengers travelling in their own vehicles. Therefore, our objective function is composed of two weighted components: the number of *accepted* passengers and the *saved distance* (i.e., the difference between the sum of direct distances between all origins and destinations and the total distance driven by our vehicles), which we want to maximize. The optimization problem now consists of deciding which passenger requests are accepted, and which are rejected, to assign accepted requests to one of the $\kappa$ vehicles, and to plan the routes of these vehicles.

The above-defined problem is a variant of the Dial-a-Ride problem: removing the restriction that vehicles may only turn without passengers on board, it reduces to a (standard) DARP. Given that our vehicle's operations are constrained by the line, we call our problem the line-based Dial-a-Ride problem (liDARP).

## 4     MILP Formulations for the liDARP

The underlying line structure, which defines an order of bus stops, combined with the directionality property, allows us to divide the route of each vehicle into a number of *sublines*: a sequence of stops at which a vehicle stops to pick-up or drop-off passengers. The first subline is initialized when a vehicle starts its route and a new subline starts after each of the vehicle's turns. We split the set of sublines $S$ into *ascending sublines* ($S^{\mathrm{asc}}$), travelling from a stop $i$ to $j$ with $i < j$, and *descending sublines* ($S^{\mathrm{desc}}$), travelling in the opposite direction.

Similarly, we divide the passenger requests $r \in R$ into *ascending requests* ($R^{\mathrm{asc}}$) and *descending requests* ($R^{\mathrm{desc}}$). As passengers may not be on board when the vehicle turns, each accepted request can be assigned to exactly one subline, with ascending requests assigned to ascending sublines and descending requests assigned to descending sublines.

In Section 4.1, we exploit these properties to propose a *Subline-Based* MILP for the liDARP, explicitly modelling sublines and passenger assignments to sublines. In Section 4.2 and Section 4.3, we show that the sublines can also be used to simplify MILP formulations for the standard DARP.

Note that the sublines in the liDARP are similar to the so-called *fragments* proposed by Rist and Forbes [28] in their branch-and-cut approach for the DARP. Namely, a subline can be further subdivided into fragments, which start with a pick-up node and end when the vehicle is empty.

### 4.1     Subline-Based Formulation

The *Subline-Based formulation* relies on the concept of a subline. Given the set of vehicles $K$, we assign each $k \in K$ a set of sublines $S$ and use binary variables $y_i^{s,k}$ to indicate whether subline $s$ of vehicle $k$ stops at bus stop $i$. The route of every subline $s$ of vehicle $k$ is encoded by the binary variables $x_{i,j}^{s,k}$ that denote the path between bus stops $i,j \in H$. Depending on the direction of the subline $s$, these only need to be defined for $i \leq j$ or $j \leq i$, respectively. Sublines are computed on a vehicle-basis and symmetry breaking constraints are defined on the vehicle's index to remove alternative solutions with equal objectives.

Flow conservation constraints ensure these routes are consistent in each subline and between consecutive sublines. We track the start and end stop of each subline with binary variables $x_{i,i}^{s,k}$, which correspond to sublines $s$ of vehicle $k$ turning at bus stop $i$.

Requests are assigned to sublines using binary variables $\mathrm{assign}_r^{s,k}$ to indicate if request $r$ is transported by subline $s$ of vehicle $k$. Note that these only need to be created for pairs $(r, s)$ with both $r$ and $s$ travelling in the same direction. We ensure each passenger is picked up at most once, with the corresponding subline stopping at both the origin and destination stop. The underlying line structure determines each subline's pick-up and drop-off sequence, allowing capacity constraints to be expressed solely in variables $\mathrm{assign}_r^{s,k}$.

Continuous variables $\mathrm{arr}_i^{s,k}$ and $\mathrm{dep}_i^{s,k}$ model the arrival and departure time of subline $s$ of vehicle $k$ at bus stop $i$, respectively. We introduce constraints to ensure the stopping time is sufficiently long for all assigned passengers who are boarding or alighting the vehicle at a station to do so. Similarly, we ensure that the time between departure at a bus stop $i$ and arrival at the next bus stop $j$ on the vehicle's route is equal to $t_{i,j}$. The departure and arrival times are further constrained by the fact that, when a request is assigned to a subline, the corresponding departure and arrival times have to respect the request's time window and associated service promises. For this, we track the pick-up and arrival time of every request $r$.

An overview of parameters and variables, and the full model are given in Appendix B.1.

## 4.2   Location-Based Formulation

Inspired by mathematical programming formulation for the traveling salesperson and vehicle routing problems, Cordeau [7] models the Dial-a-Ride problem using a graph where nodes represent origin $o_r$ and destination $d_r$ locations of requests $r$ and arcs represent direct connections between locations. In principle, traveling between any pair of locations is possible, though many arcs can be removed in a pre-processing step based on time constraints.

To account for the directionality property, we can modify Cordeau's DARP formulation for use in the liDARP as follows: we treat the requests as (general) DARP input, adding modifications to respect the line precedence and to prevent that vehicles turn with passengers on board. Observing that a vehicle may only turn after drop-off location or before a pick-up location, we introduce additional nodes at these bus stops which are used to start or end a turn, allowing us to model our problem based on fewer arcs than the general DARP. A *start-turn* node denotes that a vehicle is turning at a pick-up node (and then starting a new subline), while an *end-turn* node denotes that a vehicle is turning after a drop-off node (and ending the current subline). Similar to the general DARP, many arcs can be excluded by pre-processing based on time windows and service constraints.

Binary variables $x_{i,j}^k$ model whether vehicle $k$ travels from node $i$ to node $j$ with flow constraints ensuring feasible operations. Additional variables and big-$M$-constraints are needed to keep track of the time at which locations are visited and of vehicle load, so that the requirements with respect to time windows, service promises, and vehicle capacity are ensured. Technical details and the full model are given in Appendix B.2, where we use strengthening techniques based on [11].

## 4.3   Event-Based Formulation

Encoding feasible user allocations in vehicles as nodes, and constructing only edges between feasible connections, Gaul et al. [14] propose the *Event-Based* graph as a basis to formulate the general DARP as a MILP. Every node in the Event-Based graph represents a $Q_{\mathrm{max}}$-tuple,

where the first entry represents the most recent action: a pick-up $r^+$ or drop-off $r^-$ of a request $r$, and the remaining entries denote the other passengers on board. The node $\mathbf{0}$ is used to denote the depot. Finding feasible vehicle routes for the DARP can then be interpreted as a minimum cost circulation flow problem with the additional constraints that each passenger cannot be picked up more than once and that time windows and service constraints need to be respected.

While the number of events grows exponential with $Q_{\max}$, many nodes can already be excluded during the construction of the Event-Based graph due to incompatibility of time windows and service constraints. The directionality property allows us to further reduce this set. In particular, requests $i$ and $j$ cannot be part of the same event if

- one of them is ascending and the other is descending, or
- both requests are ascending (resp. descending) and the request with a later (resp. earlier) starting station cannot board a vehicle with the other already on board due to time window constraints.

The Event-Based model for the DARP can be directly applied to the liDARP by an adapted construction of the Event-Based graph, distinguishing ascending and descending events and connecting only events that preserve the directionality property, hence we do not re-state the formulation here. Moreover, Gaul et al. [12] proposed further arc eliminations.

## 5     Computational Experiments

In this section, we present numerical experiments for the liDARP on synthetic benchmark instances. For all experiments, we set the passenger load $q_r = 1$ and service time $b_r = 3\,\text{min}$ for all $r \in R$. The service promise parameters were set to a maximum waiting time of $\beta = 15\,\text{min}$ and a maximum exceedance of direct ride time by $\alpha = 3$. The objective function weights were chosen to be $c_1 = 10$ for the number of accepted passenger and $c_2 = 1$ for the saved distance for the computational results.

The models for the Subline-Based and Location-Based formulation were implemented in Python 3.11 using Gurobi 10.0. The Event-Based formulation was implemented in C++ 17 using CPLEX 22.1, based on the code by Gaul et al. [13]. The computations are carried out using a 12th Gen Intel Core i7-1260P CPU, running at 2.10 GHz with 32 GB RAM. For all runs, we set the solver timeout to 60 min and repeated the calculation five times, averaging the runtimes.

### 5.1     Benchmark Instances

We create new benchmark instances specifically for the liDARP using the existing bus stops of bus line 6 in Würzburg, Germany, as pictures in Figure 1, with 16 stops connecting the city center to a residential area. We calculate bus stop distances using OpenStreetMap [21], assuming vehicles can take shortcuts and rounding to the nearest minute.

For the given bus stops $H$, we generate requests that uniformly choose a pick-up and drop-off stop in $H$. We generate an equal amount of requests per time window type, picking the (earliest) pick-up time or (latest) drop-off time uniformly in the interval $[0, 480]$, corresponding to an operation of 8 hours. The vehicles have a capacity of $Q_{\max} \in \{3, 6\}$ and take $t_{\text{turn}} = 3\,\text{min}$ to turn.

We generated 14 instances on the given sequence $H$, varying from 16 requests with 2 vehicles to 50 requests with 5 vehicles, following the sizing of the well-known benchmark instances by Cordeau [7] for the classical DARP. The instance names consist of a prefix 'w'

**Figure 1** Route of bus line 6 in Würzburg, Germany, from [21].

(for Würzburg), followed by two numbers, where the first indicates the number of vehicles and the second denotes the number of requests.

## 5.2 Results

In this section, we first compare the three proposed MILP formulations using the benchmark bus line test instances. Second, we assess the trade-off between environmental savings and customer satisfaction. Lastly, we compare the liDARP model to the classical DARP model to evaluate its competitiveness.

### Computational Time

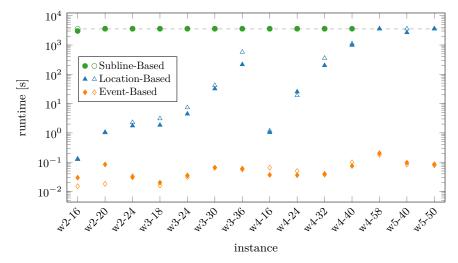sFigure 2 shows the computational time per benchmark instance for all three formulations.



**Figure 2** Runtime of the three formulations on the benchmark instances. Solid markers denote $Q_{\max} = 3$, unfilled markers $Q_{\max} = 6$, and the dashed line marks the solver timeout.

The results clearly show that the Event-Based model outperforms the Location-Based and

Subline-Based models in all instances, for either capacity. For $Q_{max} = 3$, the Subline-Based model reached the timeout for all but one instance, w2-16, and we were not able to compute a solution for the three largest instances due a lack of available memory. For $Q_{max} = 6$, the largest possible instance the model could solve was w4-32.

Observing the Location-Based model, we see a step-structure, where the runtime significantly increases with the number of requests, then decreases as the instances switch to the next-largest number of vehicles. We see a similar effect in the Event-Based model between instances w4-58 and w5-40. This is supported by the model size differences reported in Table 1, which strongly correlate to the number of passenger requests.

In the two instances where the Location-Based model reached timeout for $Q_{max} = 3$, namely w4-58 and w5-50, we note that although the achieved a relative MIP gap at timeout was greater than 1, the objective value found was within 8 % and even 0 % of the optimum, respectively.

■ **Table 1** Number of constraints and variables for the benchmark test instances with $Q_{max} = 3$, ordered by the number of requests. SB = Subline-Based, LB = Location-Based, EB = Event-Based.

| | Num. Constraints | | | Num. Boolean Var. | | | Num. Cont. Var. | | |
|---|---|---|---|---|---|---|---|---|---|
| Inst. | SB | LB | EB | SB | LB | EB | SB | LB | EB |
| w2-16 | 45 042 | 553 | 317 | 20 030 | 968 | 189 | 1952 | 150 | 53 |
| w4-16 | 90 388 | 787 | 335 | 40 188 | 1868 | 196 | 3872 | 154 | 56 |
| w3-18 | 83 315 | 752 | 401 | 35 841 | 1731 | 238 | 3276 | 170 | 64 |
| w2-20 | 67 290 | 689 | 497 | 28 062 | 1372 | 294 | 2440 | 186 | 76 |
| w2-24 | 96 434 | 825 | 746 | 38 462 | 2062 | 430 | 2928 | 222 | 106 |
| w3-24 | 144 443 | 998 | 830 | 57 117 | 3168 | 449 | 4368 | 224 | 119 |
| w4-24 | 197 732 | 1171 | 769 | 79 228 | 4116 | 437 | 5808 | 226 | 109 |
| w3-30 | 227 963 | 1244 | 956 | 87 213 | 4473 | 610 | 5460 | 278 | 122 |
| w4-32 | 352 756 | 1555 | 1174 | 133 500 | 6792 | 706 | 7744 | 298 | 149 |
| w3-36 | 340 895 | 1490 | 1627 | 127 749 | 6564 | 935 | 6552 | 332 | 199 |
| w4-40 | 577 636 | 1939 | 1867 | 214 204 | 10 668 | 1117 | 9680 | 370 | 223 |
| w5-40 | 722 789 | 2224 | 1657 | 267 755 | 12 945 | 1065 | 12 080 | 372 | 188 |
| w5-50 | 1 227 649 | 2774 | 2419 | 453 155 | 20 885 | 1602 | 15 100 | 462 | 249 |
| w4-58 | 1 400 268 | 2803 | 3282 | 517 020 | 21 976 | 2154 | 14 036 | 532 | 326 |

The Subline-Based model requires a significantly higher number of resources, with the number of constraints and boolean variables exceeding those of both the Location-Based and Event-Based models by factors of 100 and 10, respectively. Notably, the Location-Based model uses 10 times more boolean variables than the Event-Based model, but both require a similar amount of continuous variables.

In general, this result is not surprising, as Gaul et al. [14] demonstrated the computational efficiency of the Event-Based graph in a MILP formulation for the classical DARP, as many complicating constraints are implicitly encoded in the underlying network structure.

All following experiments are carried out with $Q_{max} = 3$.

## Trade-off Analysis

To evaluate the trade-off between environmental savings and customer attractiveness in our chosen objective function, we compare three different settings: in the *environmentally focused* setting, we use objective weights $c_1 = 1$, $c_2 = 10$, placing an emphasis on the distance saved,

whilst in the *customer focused* setting, we use weights $c_1 = 10, c_2 = 1$, emphasising the number of transported passengers. We also include a setting with equal weights as a base case. The trade-off between objective function components is visualized in Figure 3.
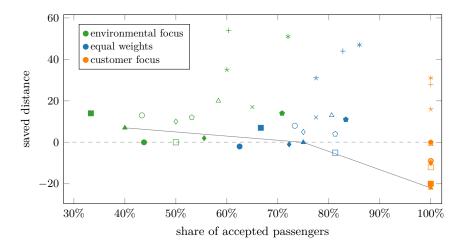


**Figure 3** Objective Function parameters for varying weights. Each marker shape represents one benchmark instance.

Note that both objective functions are considered as maximizing objectives. A positive saved distance is preferred, as this corresponds to more direct passenger kilometers saved than total routing costs accumulated. All instances in Figure 3 were solved to optimality.

We observe that, in the customer-focused setting, all passengers are accepted in all instances. In the environmental-focused setting, the saved distance is always non-negative. We have highlighted the approximated Pareto front for a specific instance, w2-20, represented by triangles, by connecting the markers corresponding to the three obtained weighted-sum solutions in Figure 3 to better illustrate the trade-offs. The saved distance decreases from 7 min to −22 min between the environmental-focused and the customer-focused setting, while the share of accepted passengers increases from 40 % to 100 %. The same pattern can be observed for all other benchmark instances. Hence, the proposed objective function is capable of capturing multiple needs and can be adjusted accordingly, dependent on the chosen application.

## DARP versus liDARP

Lastly, we compare the liDARP formulation to the classical DARP formulation, where vehicles are allowed to take any route between passenger without needing to adhere to the line structure. We use the above-introduced benchmark instances, extending these to cases with up to 11 vehicles and 132 requests (similar to the extended benchmark set introduced by Røpke et al. [29]). We set $t_{\text{turn}} = 0\,\text{min}$ for both formulations and set the solver timeout to 60 min. The computational time for both models is shown in Figure 4, averaged over five runs. The Event-Based model was used to produce the liDARP results.

We observe that the liDARP model is faster in all instances. While both models' computational time increases with the number of requests, the liDARP was able to solve even the largest instances with over 100 requests in less than 10 s, while the DARP model was aborted at timeout. Examining the objective values, both models accepted all requests in all instances, while they differ in the saved distance, which is visualized in Figure 5. The
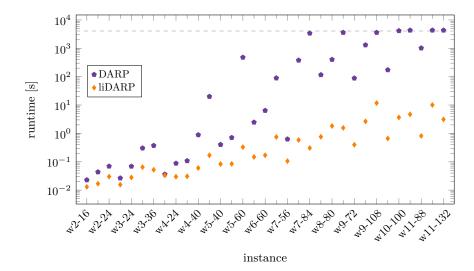
**Figure 4** Runtime of the DARP and liDARP on the extended benchmark instances. The dashed line marks the solver timeout.

DARP achieved a higher saved distance in all but one instance, w10-100, which was aborted at timeout, with the average deviation being 3 min and the maximum deviation being a saving of 10 min in instance w11-132. Both models use all available vehicles for all instances.
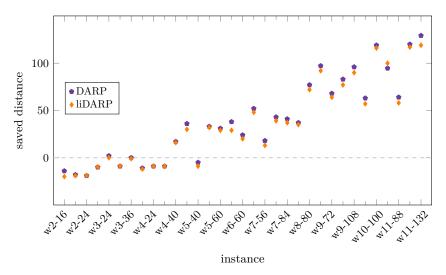


**Figure 5** Saved distance of the liDARP and DARP on the extended benchmark instances.

The average ride time, measured as the time between pick-up and drop-off of each request, was marginally higher in the DARP model, with an average increase of 0.14 min. The difference in the average share of empty mileage, defined as the fraction of empty mileage over the total distance, is less than 0.02 on average. Similarly, the difference in average detour, which is the fraction of passenger distance driven to the shortest distances between pick-up and drop-off, is less than 0.025 on average.

Examining each DARP solution, we count the number of requests which are, at least for a portion of their trip, travelling *away* from their destination, i.e., violating the directionality property. The only instances where there are no such violations are w3-18, w3-30, and w4-32.

On average, 7.7 % of passengers travel in the opposite direction for at least a portion of their trip, with the largest amount being 18.9 % in instance w9-90. We hypothesize that these routes will likely be viewed as unnecessary by customers, even if they are the most efficient amongst all possible connections.

While there is a difference in saved distance and ride time, passengers do not have to accept significant detours when using the liDARP compared to the DARP model, and the vehicles travel without passengers for a similar amount of time.

## 6    Conclusion

We present the line-based Dial-a-Ride problem (liDARP), wherein ridepooling vehicles operate on-demand on a sequence of bus stations, adhering to the directionality property, time and capacity constraints, and our service promises. The efficiency of this approach is validated through numerical computations on benchmark instances derived from on a real-life bus line in Würzburg, Germany. Our multi-objective approach successfully balances environmental concerns, by reducing the total distance travelled compared to individual passenger trips, and the attractiveness, measured in the total amount of passengers accepted.

The advantages of the liDARP additionally include the possibility to use existing infrastructure, being based on a sequence of bus stops, which may increase utilization in off-peak times or in areas with low demand. Thus, we provide a system that can serve as an alternative to public transport, providing flexibility to its customers to improve attractiveness.

Our approach can be extended to inhomogeneous vehicles with varying capacities, to determine where each vehicle is best allocated to serve the customer base. Future research could explore varying demand scenarios, such as operating a feeder line to a train station or considering rush-hour induced fluctuations in demand. Finally, this paper focuses on the static variant of the liDARP, where all requests are known in advance. Investigating the liDARP under unknown and dynamic demand may provide insights into its practical applicability and competitiveness in a real-world setting.

─── **References** ───

1    Dilay Aktas, Pieter Vansteenwegen, and Kenneth Sörensen. A demand-responsive bus system for peak hours with capacitated vehicles. In *Proc. 11th Triennial Symposium on Transportation Analysis conference (TRISTAN XI)*, Mauritius Island, 2022. TRISTAN.
2    Claudia Archetti, Dominique Feillet, Michel Gendreau, and M. Grazia Speranza. Complexity of the VRP and SDVRP. *Transportation Research Part C: Emerging Technologies*, 19(5):741–750, August 2011. `doi:10.1016/j.trc.2009.12.006`.
3    Andrea Attanasio, Jean-François Cordeau, Gianpaolo Ghiani, and Gilbert Laporte. Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. *Parallel Computing*, 30(3):377–387, March 2004. `doi:10.1016/j.parco.2003.12.001`.
4    John W. Baugh, Gopala Krishna Reddy Kakivaza, and John R. Stone. Intractability of the Dial-a-Ride Problem and a Multiobjective Solution Using Simulated Annealing. *Engineering Optimization*, 30(2):91–123, February 1998. `doi:10.1080/03052159808941240`.
5    Antje Bjelde, Jan Hackfeld, Yann Disser, Christoph Hansknecht, Maarten Lipmann, Julie Meißner, Miriam Schlöter, Kevin Schewior, and Leen Stougie. Tight bounds for online tsp on the line. *ACM Transactions on Algorithms (TALG)*, 17(1):1–58, 2021. `doi:10.1145/3422362`.
6    Ingrid Busch. *Vehicle routing on acyclic networks*. Dissertation, The Johns Hopkins University, Baltimore, Maryland, 1991.
7    Jean-François Cordeau. A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research*, 54(3):573–586, 2006. `doi:10.1287/opre.1060.0283`.

**8** Jean-François Cordeau and Gilbert Laporte. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6):579–594, 2003. `doi:10.1016/S0191-2615(02)00045-0`.

**9** Jean-François Cordeau and Gilbert Laporte. The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, 153(1):29–46, 2007. `doi:10.1007/s10479-007-0170-8`.

**10** Willem E. de Paepe, Jan Karel Lenstra, Jiri Sgall, René A. Sitters, and Leen Stougie. Computer-aided complexity classification of dial-a-ride problems. *INFORMS Journal on Computing*, 16(2):120–132, 2004. `doi:10.1287/ijoc.1030.0052`.

**11** Martin Desrochers and Gilbert Laporte. Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters*, 10(1):27–36, 1991. `doi:10.1016/0167-6377(91)90083-2`.

**12** Daniela Gaul, Kathrin Klamroth, Christian Pfeiffer, Arne Schulz, and Michael Stiglmayr. A Tight Formulation for the Dial-a-Ride Problem, 2023. `arXiv:2308.11285`.

**13** Daniela Gaul, Kathrin Klamroth, and Michael Stiglmayr. Solving the Dynamic Dial-a-Ride Problem Using a Rolling-Horizon Event-Based Graph. In *21st Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2021)*, volume 96 of *Open Access Series in Informatics (OASIcs)*, pages 8:1–8:16, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/OASIcs.ATMOS.2021.8`.

**14** Daniela Gaul, Kathrin Klamroth, and Michael Stiglmayr. Event-based MILP models for ridepooling applications. *European Journal of Operational Research*, 301(3):1048–1063, 2022. `doi:10.1016/j.ejor.2021.11.053`.

**15** Konstantinos Gkiotsalitis, Marie Schmidt, and Evelien van der Hurk. Subline frequency setting for autonomous minibusses under demand uncertainty. *Transportation Research Part C: Emerging Technologies*, 135:103492, 2022. `doi:10.1016/j.trc.2021.103492`.

**16** Timo Gschwind and Stefan Irnich. Effective handling of dynamic time windows and its application to solving the dial-a-ride problem. *Transportation Science*, 49(2):335–354, 2015. `doi:10.1287/trsc.2014.0531`.

**17** Sin C. Ho, Wai Yuen Szeto, Yong-Hong Kuo, Janny M.Y. Leung, Matthew Petering, and Terence W.H. Tou. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, 111:395–421, 2018. `doi:10.1016/j.trb.2018.02.001`.

**18** Omar J. Ibarra-Rojas, Felipe Delgado, Ricardo Giesen, and Juan Carlos Muñoz. Planning, operation, and control of bus transport systems: A literature review. *Transportation Research Part B: Methodological*, 77:38–75, 2015. `doi:10.1016/j.trb.2015.03.002`.

**19** Pei Liu, Marie Schmidt, Qingxia Kong, Joris Camiel Wagenaar, Lixing Yang, Ziyou Gao, and Housheng Zhou. A robust and energy-efficient train timetable for the subway system. *Transportation Research Part C: Emerging Technologies*, 121:102822, 2020. `doi:10.1016/j.trc.2020.102822`.

**20** Yves Molenbruch, Kris Braekers, and An Caris. Typology and literature review for dial-a-ride problems. *Annals of Operations Research*, 259(1):295–325, 2017. `doi:10.1007/s10479-017-2525-0`.

**21** OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

**22** Sophie N. Parragh. Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. *Transportation Research Part C: Emerging Technologies*, 19(5):912–930, 2011. `doi:10.1016/j.trc.2010.06.002`.

**23** Sophie N. Parragh and Verena Schmid. Hybrid column generation and large neighborhood search for the dial-a-ride problem. *Computers & Operations Research*, 40(1):490–497, 2013. `doi:10.1016/j.cor.2012.08.004`.

**24** Christian Pfeiffer and Arne Schulz. An ALNS algorithm for the static dial-a-ride problem with ride and waiting time minimization. *OR Spectrum*, 44(1):87–119, 2022. `doi:10.1007/s00291-021-00656-7`.

**25**    Harilaos N. Psaraftis. A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science*, 14(2):130–154, 1980. `doi:10.1287/trsc.14.2.130`.

**26**    Harilaos N. Psaraftis. An Exact Algorithm for the Single Vehicle Many-to-Many Dial-A-Ride Problem with Time Windows. *Transportation Science*, 17(3):351–357, 1983. `doi:10.1287/trsc.17.3.351`.

**27**    Line Blander Reinhardt, Tommy Clausen, and David Pisinger. Synchronized dial-a-ride transportation of disabled passengers at airports. *European Journal of Operational Research*, 225(1):106–117, 2013. `doi:10.1016/j.ejor.2012.09.008`.

**28**    Yannik Rist and Michael A. Forbes. A New Formulation for the Dial-a-Ride Problem. *Transportation Science*, 55(5):1113–1135, 2021. `doi:10.1287/trsc.2021.1044`.

**29**    Stefan Ropke, Jean-François Cordeau, and Gilbert Laporte. Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks*, 49(4):258–272, 2007. `doi:10.1002/net.20177`.

**30**    Stefan Ropke and David Pisinger. An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows. *Transportation Science*, 40(4):455–472, 2006. `doi:10.1287/trsc.1050.0135`.

## A    Variable Overview

**Table 2** Summary of liDARP parameters.

| Notation | Definition |
|---|---|
| $H$ | set of bus stops, $\{1, \ldots, n\}$ |
| $K$ | set of vehicles, $\{1, \ldots, \kappa\}$ |
| $R$ | set of passenger requests, $\{1, \ldots, m\}$ |
| $R^{\mathrm{asc}}$ | set of passenger requests travelling in ascending direction |
| $R^{\mathrm{desc}}$ | set of passenger requests travelling in descending direction |
| $o_r$ | origin stop of passenger request $r$ |
| $d_r$ | destination stop of passenger request $r$ |
| $q_r$ | load of request $r$ |
| $e_r$ | earliest departure time of request $r$ |
| $l_r$ | latest arrival time of request $r$ |
| $b_r$ | service time for request $r$ |
| $t_{i,j}$ | travel time from bus stop $i$ to $j$ |
| $\alpha$ | service promise constant relating to maximum ride time, excess factor |
| $\beta$ | service promise constant relating to maximum wait time |
| $L_r$ | total ride time of request $r$ |
| $L_r^{\max}$ | maximum ride time of request $r$, dependent on $\alpha$ and $\beta$ |
| $t_{\mathrm{turn}}$ | time it takes for a vehicle to turn around |
| $c_1, c_2$ | objective weights |

## B    MILP Formulations

### B.1    Subline-Based Formulation

In this section, we present the MILP model for the Subline-Based formulation introduced in Section 4.1. All parameters and variables are summarized in Table 3.

■ **Table 3** Summary of notation for the Subline-Based model.

| Notation | Definition |
|---|---|
| **Parameters** | |
| $S$ | set of sublines, $\{1, \ldots, \sigma\}$ |
| $S^{\mathrm{asc}}$ | set of sublines travelling in ascending direction |
| $S^{\mathrm{desc}}$ | set of sublines travelling in descending direction |
| **Binary Decision Variables** | |
| $\mathrm{assign}_r^{s,k}$ | 1 if request $r$ is assigned to subline $s$ of vehicle $k$ |
| $y_i^{s,k}$ | 1 if subline $s$ of vehicle $k$ visits node $i$ |
| $\mathrm{start}_i^k$ | 1 if node $i$ is the start node of vehicle $k$ |
| $\mathrm{end}_i^k$ | 1 if node $i$ is the end node of vehicle $k$ |
| $x_{i,j}^{s,k}$ | 1 if node $j$ is visited immediately after node $i$ on subline $s$ of vehicle $k$ |
| $x_{i,i}^{s,k}$ | 1 if vehicle $k$ turns at node $i$ after executing subline $s$ |
| $w_{r_1,r_2}$ | 1 if requests $r_1$ and $r_2$ are on the same subline of the same vehicle |
| $z_k$ | 1 if vehicle $k$ is in use |
| **Continuous Decision Variables** | |
| $\mathrm{dep}_i^{s,k}$ | departure time of subline $s$ of vehicle $k$ at node $i$ |
| $\mathrm{arr}_i^{s,k}$ | arrival time of subline $s$ of vehicle $k$ at node $i$ |
| $\mathrm{pickup}_r$ | pick-up time of request $r$ |
| $\mathrm{arrtime}_r$ | drop-off time of request $r$ |

In this model, we explicitly model the path of every subline of each vehicle, using binary variables $y_i^{s,k}$ to denote if subline $s$ of vehicle $k$ stops at bus station $i$ and binary variables $x_{i,j}^{s,k}$ to denote if the direct path from station $i$ to station $j$ is used. Further binary variables $\mathrm{start}_i^k$ and $\mathrm{end}_i^k$ denote if vehicle $k$ starts and ends at station $i$, respectively. Then, by tracking the turning stations of every subline $s$, i.e., stations $i$ where $x_{i,i}^{s,k} = 1$, we track the start and end station of every subline. For every variable which references both passengers and sublines, such as $\mathrm{assign}_r^{s,k}$, we only create those variables where the passenger and subline are travelling in the same direction.

To ensure our model respects the boarding precedence (requests which are alighting leave the vehicle before those boarding can enter), we construct the following three subsets for every request $i \in R$:

- $\mathrm{Prec}_i^{o,o}$, containing all requests $j$ which have the same origin as $i$, are travelling in the same direction, and should board before $i$,
- $\mathrm{Prec}_i^{o,d}$, containing all requests $j$ whose destination is at $i$'s origin, are travelling in the same direction, and should alight before $i$ boards, and
- $\mathrm{Prec}_i^{d,d}$, containing all requests $j$ which have the same destination as $i$, are travelling in the same direction, and should alight before $i$.

Note that the set $\mathrm{Prec}_i^{d,o}$, which denotes all origin requests that need to be served before request $i$ is dropped-off, is not created as it is always empty due to the boarding assumption.

Furthermore, we track which passengers are pooled together on the same subline of the same vehicle with binary variables $w_{r_1,r_2}$. These are required, together with the precedence sets, to ensure we allow for sufficient boarding and alighting times per passenger at every stop. To model these, we use big-M constraints with $M_1 := \max_{r \in R} L_r^{\mathrm{max}}$.

Binary variables $z_k$ denote that vehicle $k$ is used by our solution. Here, we use big-M constraints with $M_2 := \max_{r \in R} l_r + (2 \cdot \sigma - 1) \cdot b_r$, where $\sigma$ denotes the number of sublines, to ensure the arrival and departure time variables for each vehicle are only set if they are also used.

To strengthen the model, we enforce that a vehicle's end stop is placed after it has turned twice, i.e., after two consecutive sublines start and end at the same stop. Then, every following subline is empty and turns at the same stop. This reduces the number of possible solutions with the same objective value.

We define $T^+ := \max_{r \in R} L_r^{\max} + (\sigma - 1)t_{\text{turn}}$ to be the end of service, i.e., when all vehicles end their operation at the latest.

We note that the model's size is dependent on the choice of $\sigma$, the number of sublines, which is hard to choose. We set $\sigma = 2 \cdot m$ for all experiments presented here.

The full Subline-Based model is given by:

$$\max_x c_1 \cdot \left( \sum_{s \in S} \sum_{k \in K} \sum_{r \in R} \text{assign}_r^{s,k} \cdot t_{o_r, d_r} - \sum_{k \in K} \sum_{s \in S} \sum_{\substack{(i,j) \in H \times H: \\ i \neq j}} x_{i,j}^{s,k} \cdot t_{i,j} \right)$$

$$+ c_2 \cdot \sum_{s \in S} \sum_{k \in K} \sum_{r \in R} \text{assign}_r^{s,k}$$

$$\text{s.t.} \quad \sum_{i \in H} \text{start}_i^k \leq 1 \qquad \forall\, k \in K \tag{1a}$$

$$\sum_{i \in H} \text{end}_i^k = \sum_{i \in H} \text{start}_i^k \qquad \forall\, k \in K \tag{1b}$$

$$\sum_{i \in H} x_{ii}^{s,k} = \sum_{i \in H} \text{start}_i^k \qquad \forall\, k \in K, s \in S \setminus \{\sigma\} \tag{1c}$$

$$x_{i,i}^{\sigma,k} = 0 \qquad \forall\, i \in H, k \in K \tag{1d}$$

$$y_i^{1,k} = \text{start}_i^k + \sum_{j < i} x_{j,i}^{1,k} \qquad \forall\, i \in H, k \in K \tag{1e}$$

$$y_i^{s,k} = x_{i,i}^{s-1,k} + \sum_{j < i} x_{j,i}^{s,k} \qquad \forall\, i \in H, k \in K, s \in S^{\text{asc}} \setminus \{1\} \tag{1f}$$

$$y_i^{s,k} = x_{i,i}^{s-1,k} + \sum_{j > i} x_{j,i}^{s,k} \qquad \forall\, i \in H, k \in K, s \in S^{\text{desc}} \tag{1g}$$

$$y_i^{s,k} = x_{i,i}^{s,k} + \sum_{j > i} x_{i,j}^{s,k} \qquad \forall\, i \in H, k \in K, s \in S^{\text{asc}} \tag{1h}$$

$$y_i^{s,k} = x_{i,i}^{s,k} + \sum_{j < i} x_{i,j}^{s,k} \qquad \forall\, i \in H, k \in K, s \in S^{\text{desc}} \setminus \{\sigma\} \tag{1i}$$

$$y_i^{\sigma,k} = \text{end}_i^k + \sum_{j < i} x_{i,j}^{\sigma,k} \qquad \forall\, i \in H, k \in K \tag{1j}$$

$$z_k = \sum_{i \in H} \text{start}_i^k \qquad \forall\, k \in K \tag{1k}$$

$$\sum_{r: o_r \leq i, d_r > i} \text{assign}_r^{s,k} \leq Q_{\max} \qquad \forall\, k \in K, s \in S^{\text{asc}}, i = 1, \ldots, n-1 \tag{1l}$$

$$\sum_{r: o_r \geq i, d_r < i} \text{assign}_r^{s,k} \leq Q_{\max} \qquad \forall\, k \in K, s \in S^{\text{desc}}, i = 2, \ldots, n \tag{1m}$$

$$z_k \geq y_i^{s,k} \qquad \forall\, i \in H, s \in S, k \in K \tag{1n}$$

$$z_k \geq x_{i,j}^{s,k} \qquad \forall (i,j) \in E, s \in S, k \in K \tag{1o}$$

$$z_k \geq \mathrm{assign}_r^{s,k} \qquad \forall\, r \in R, s \in S, k \in K \tag{1p}$$

$$M_2 \cdot z_k \geq \mathrm{arr}_i^{s,k} \qquad \forall\, i \in H, s \in S, k \in K \tag{1q}$$

$$M_2 \cdot z_k \geq \mathrm{dep}_i^{s,k} \qquad \forall\, i \in H, s \in S, k \in K \tag{1r}$$

$$\mathrm{start}_i^k \geq \sum_{j \in H: j < i} \mathrm{start}_j^{k+1} \qquad \forall\, i \in H, k \in \{1, \dots, \kappa - 1\} \tag{1s}$$

$$\mathrm{dep}_i^{s,k} \geq \mathrm{arr}_i^{s,k} \qquad \forall\, i \in H, s \in S, k \in K \tag{1t}$$

$$\mathrm{arr}_j^{s,k} \geq \mathrm{dep}_i^{s,k} + t_{i,j} \cdot x_{i,j}^{s,k} \qquad \forall (i,j) \in H \times H \text{ with } i < j, s \in S^{\mathrm{asc}}, k \in K \tag{1u}$$

$$\mathrm{arr}_j^{s,k} \geq \mathrm{dep}_i^{s,k} + t_{i,j} \cdot x_{i,j}^{s,k} \qquad \forall (i,j) \in H \times H \text{ with } i > j, s \in S^{\mathrm{desc}}, k \in K \tag{1v}$$

$$\mathrm{arr}_i^{s,k} \geq \mathrm{dep}_i^{s-1,k} + t_{\mathrm{turn}} \cdot x_{i,i}^{s-1,k} \qquad \forall\, i \in H, s \in S^{\mathrm{asc}} \setminus \{1\}, k \in K \tag{1w}$$

$$\mathrm{arr}_i^{s,k} \geq \mathrm{dep}_i^{s-1,k} + t_{\mathrm{turn}} \cdot x_{i,i}^{s-1,k} \qquad \forall\, i \in H, s \in S^{\mathrm{desc}}, k \in K \tag{1x}$$

$$\mathrm{arr}_i^{s,k} \geq 0 \qquad \forall\, i \in H, s \in S, k \in K \tag{1y}$$

$$\sum_{k \in K} \sum_{s \in S} \mathrm{assign}_r^{s,k} \leq 1 \qquad \forall\, r \in R \tag{1z}$$

$$2 \cdot \mathrm{assign}_r^{s,k} \leq y_{o_r}^{s,k} + y_{d_r}^{s,k} \qquad \forall\, k \in K, s \in S, r \in R \tag{1aa}$$

$$\mathrm{assign}_r^{s,k} \cdot (e_r + b_{o_r}) \leq \mathrm{dep}_{o_r}^{s,k} \qquad \forall\, k \in K, s \in S, r \in R \tag{1ab}$$

$$l_r + T^+ \cdot (1 - \mathrm{assign}_r^{s,k}) \geq \mathrm{arr}_{d_r}^{s,k} \qquad \forall\, r \in R, s \in S, k \in K \tag{1ac}$$

$$w_{r_i,r_j} \leq \mathrm{assign}_{r_i}^{s,k} \qquad \forall\, s \in S, k \in K, r_i, r_j \in R \tag{1ad}$$

$$\mathrm{assign}_{r_i}^{s,k} + \mathrm{assign}_{r_j}^{s,k} - 1 \leq w_{r_i,r_j} \qquad \forall\, s \in S, k \in K, r_i, r_j \in R \tag{1ae}$$

$$\mathrm{pickup}_r \geq \sum_{k \in K} \sum_{s \in S} \mathrm{assign}_r^{s,k} \cdot \mathrm{arr}_{o_r}^{s,k} \qquad \forall\, k \in K, s \in S, r \in R \tag{1af}$$

$$\mathrm{arrtime}_r \geq \sum_{k \in K} \sum_{s \in S} \mathrm{assign}_r^{s,k} \cdot \mathrm{arr}_{d_r}^{s,k} \qquad \forall\, k \in K, s \in S, r \in R \tag{1ag}$$

$$b_{r_i} + \mathrm{pickup}_{r_i} - \mathrm{pickup}_{r_j} \leq M_1 \cdot \left(1 - \sum_{k \in K} \sum_{s \in S} w_{r_i,r_j}^{s,k}\right) \forall r_i \in R, r_j \in \mathrm{Prec}_{r_i}^{o,o} \tag{1ah}$$

$$b_{r_i} + \mathrm{arrtime}_{r_i} - \mathrm{pickup}_{r_j} \leq M_1 \cdot \left(1 - \sum_{k \in K} \sum_{s \in S} w_{r_i,r_j}^{s,k}\right) \forall r_i \in R, r_j \in \mathrm{Prec}_{r_i}^{o,d} \tag{1ai}$$

$$b_{r_i} + \mathrm{arrtime}_{r_i} - \mathrm{arrtime}_{r_j} \leq M_1 \cdot \left(1 - \sum_{k \in K} \sum_{s \in S} w_{r_i,r_j}^{s,k}\right) \forall r_i \in R, r_j \in \mathrm{Prec}_{r_i}^{d,d} \tag{1aj}$$

$$\mathrm{dep}_{o_r}^{s,k} \geq \mathrm{assign}_r^{s,k} \cdot (\mathrm{pickup}_r + b_r) \qquad \forall\, r \in R \tag{1ak}$$

$$\mathrm{dep}_{d_r}^{s,k} \geq \mathrm{assign}_r^{s,k} \cdot (\mathrm{arrtime}_r + b_r) \qquad \forall\, r \in R \tag{1al}$$

$$e_{o_r} \leq \mathrm{pickup}_r \leq l_{o_r} \qquad \forall\, r \in R \tag{1am}$$

$$e_{d_r} \leq \mathrm{arrtime}_r \leq l_{d_r} \qquad \forall\, r \in R \tag{1an}$$

$$\mathrm{arrtime}_r + b_r - \mathrm{pickup}_r \leq \alpha \cdot t_{o_r,d_r} \qquad \forall\, r \in R \tag{1ao}$$

Constraint (1a) ensures each vehicle is used at most once and constraint (1b) ensures that a started vehicle also ends at a bus stop. Constraint (1c) makes sure every started vehicle turns around at some station, while (1d) denotes that the last subline of each vehicle does not turn. Constraints (1e) to (1i) are for flow conservation between sublines and their turn stops. The last subline ends at its end stop, which is controlled by (1j). Constraint (1k) tracks

the amount of vehicles which are used. The upper capacity of every vehicle is ensured by (1l) and (1m), for both subline directions. Constraints (1n) to (1r) ensure that only vehicles which are started can travel to and between stops. Constraint (1s) is a symmetry breaking constraint which says that vehicles with a smaller index start at a smaller station.

Constraint (1t) ensures a vehicle departs from a bus stop only after it has arrived, while constraints (1u) and (1v) ensure the arrival time at the next bus stop respects the minimum travel time. Constraints (1w) and (1x) take into account the turning time, linking a subline's end time with the subsequent subline's start time at the same stop. Finally, constraints (1y) ensures all times are positive.

Constraint (1z) ensures a passenger is picked up at most once and, by constraint (1aa), only if the subline they are assigned to also stops at their origin and destination. The link between earliest pick-up and departure from the origin times, as well as latest drop-off and arrival at the destination times, is handled with constraints (1ab) and (1ac), respectively. Finally, constraints (1ad) and (1ae) link the variable $w_{r_i,r_j}$ to denote if two passengers are assigned to the same subline of the same vehicle.

Constraints (1af) and (1ag) place a lower bound on the pick-up time and drop-off time of each passenger, dependent on the vehicle's arrival time at the corresponding station. Constraints (1ah) to (1aj) ensure the precedence rules for boarding are respected and add sufficient service times between serving customers. Then, constraints (1ak) and (1al) ensures the vehicle can only depart after the last passenger has fully boarded or alighted. The time windows of each passenger is guaranteed by (1am) and (1an), while the maximum travel time is limited by constraint (1ao).

## B.2  Location-Based Formulation

In this section, we describe the construction of the underlying graph for the Location-Based formulation in more detail, as well as presenting the full MILP model. All notation is summarized in Table 4.

We introduce *ascending* bus stops $H^{asc} := \{h_1, \ldots, h_n\}$ and *descending* bus stops $H^{desc} := \{h_{n+1}, \ldots, h_{2n}\}$. Bus stops $h_i \in H^{desc}$ and $h_{n+i} \in H^{asc}$ are virtual copies of stop $i \in H$. In reality, these may be the same stop on opposite sides of the road, for vehicles travelling in opposite directions. We connect all bus stops $h_i \in H^{desc}$ with their corresponding $h_{n+i} \in H^{asc}$, in both directions (corresponding to a turn at station $i$), and enforce that two stops in either set can only be served in upstream order with respect to the line.

Similar to the classical DARP formulation, for each request $r \in R$, we construct four nodes $o_r, d_r, \bar{o}_r, \bar{d}_r$ in a liDARP-Graph $G_R = (H_R, E_R)$. Here, the nodes $o_r$ and $d_r$ correspond to the classic pick-up and drop-off nodes of $r$. The node $\bar{o}_r$ is a *start-turn* node, denoting that the vehicle is at $o_r$, facing the opposite direction, i.e., before it turns and picks up request $r$ at $o_r$. Similarly, the node $\bar{d}_r$ is an *end-turn* node, denoting that the vehicle is at $d_r$, has dropped off the request $r$, and is now turning to continue in the opposite direction. Then,

- if $r \in R^{asc}$: we construct $o_r$ at the origin $h_{o_r} \in H^{asc}$, $d_r$ at the destination $h_{d_r} \in H^{asc}$ stop, $\bar{o}_r$ at $h_{n+o_r} \in H^{desc}$, and $\bar{d}_r$ at $h_{n+d_r} \in H^{desc}$.
- if $r \in R^{desc}$: we construct $o_r$ at the origin $h_{o_r} \in H^{desc}$, $d_r$ at the destination $h_{d_r} \in H^{desc}$ stop, $\bar{o}_r$ at $h_{o_r-n} \in H^{asc}$, and $\bar{d}_i$ at $h_{d_r-n} \in H^{asc}$.

We set $P := \{o_r : r \in R\}$, $D := \{d_r : r \in R\}$, $\bar{P} := \{\bar{o}_r : r \in R\}$, and $\bar{D} := \{\bar{d}_r : r \in R\}$ to denote the sets of these bus stops. We additionally define time windows on the nodes $\bar{o}_r$ and $\bar{d}_r$ of every request $r \in R$, dependent on the time windows on $o_r$ and $d_r$, respectively, accounting for the boarding and turn times.

| Notation | Definition |
|---|---|
| **Parameters** | |
| $\delta_{\text{start}}$ | start depot |
| $\delta_{\text{end}}$ | end depot |
| $P$ | set of pick-up nodes |
| $D$ | set of delivery node |
| $\bar{P}$ | set of start-turn nodes, before a pick-up node |
| $\bar{D}$ | set of end-turn nodes, after a drop-off node |
| $N_R$ | set of all pick-up and delivery nodes in all directions, depending on requests $R$ |
| $H_R$ | set of all pick-up, delivery, and depot nodes, depending on requests $R$ |
| $E_R$ | set of all edges between nodes in $E_R$ |
| **Binary Decision Variables** | |
| $x_{i,j}^k$ | 1 if vehicle $k$ travels on arc $(i,j) \in E_R$ |
| $z_k$ | 1 if vehicle $k$ is used |
| **Continuous Decision Variables** | |
| $B_i$ | start of service time at bus stop $i$ |
| $Q_i$ | passenger load departing bus stop $i$ |
| $L_r$ | ride time of passenger $r$ |

We introduce two depots, the *start depot* $\delta_{\text{start}}$ and the *end depot* $\delta_{\text{end}}$, where vehicles start and end their route. Let $N_R := P \cup D \cup \bar{P} \cup \bar{D}$ denote all pick-up, drop-off, and turn stops, and $H_R := \{\delta_{\text{start}}, \delta_{\text{end}}\} \cup N_R$ all nodes including the depots.

The edge set $E_R := \bigcup_{i=1}^{10} E_R^i$ between nodes in $H_R$ is constructed as follows:

- $E_R^1 := \{(o_r, d_r) \in P \times D : r \in R\}$, connecting each request's origin with its destination,
- $E_R^2 := \{(v_i, w_j) \in (P \cup D \cup \bar{D}) \times (P \cup D \cup \bar{P}) : h_{v_i}, h_{w_j} \in H^{\text{asc}}, i \neq j, v_i \text{ precedes } w_j, i, j \in R\}$, connecting all *ascending* stops to subsequent stops in the same direction,
- $E_R^3 := \{(v_i, w_j) \in (P \cup D \cup \bar{D}) \times (P \cup D \cup \bar{P}) : h_{v_i}, h_{w_j} \in H^{\text{desc}}, i \neq j, v_i \text{ precedes } w_j, i, j \in R\}$, connecting all *descending* stops to subsequent stops in the same direction,
- $E_R^4 := \{(\bar{o}_r, o_r) \in \bar{P} \times P : r \in R\}$, connecting the start-turn stop of each request with its corresponding origin stop in the opposite direction,
- $E_R^5 := \{(d_r, \bar{d}_r) \in D \times \bar{D} : r \in R\}$, connecting the destination stop of each request with its corresponding end-turn stop in the opposite direction,
- $E_R^6 := \{(v_i, w_j) \in (P \cup D \cup \bar{D}) \times (P \cup D \cup \bar{P}) : v_i, w_j \in H^{\text{asc}}, h_{v_i} = h_{w_j}, i \neq j, \neg(v_i \in D \wedge w_j \in P), e_{v_i} \leq e_{w_j}, l_{v_i} \leq l_{w_j}, i, j \in R\}$, connecting *ascending* stops at the same original physical bus stop if they are compatible regarding their time windows,
- $E_R^7 := \{(v_i, w_j) \in (P \cup D \cup \bar{D}) \times (P \cup D \cup \bar{P}) : v_i, w_j \in H^{\text{desc}}, h_{v_i} = h_{w_j}, i \neq j, \neg(v_i \in P \wedge w_j \in D), e_{v_i} \leq e_{w_j}, l_{v_i} \leq l_{w_j}, i, j \in R\}$, connecting *descending* stops at the same original physical bus stop if they are compatible regarding their time windows,
- $E_R^8 := \{(\delta_{\text{start}}, \delta_{\text{end}})\}$, connecting the starting depot to the ending depot to allow for unused vehicles,
- $E_R^9 := \{(\delta_{\text{start}}, o_r) \in \{\delta_{\text{start}}\} \times P : r \in R\} \cup \{(\delta_{\text{start}}, \bar{o}_r) \in \{\delta_{\text{start}}\} \times \bar{P} : r \in R\}$, connecting the start depot to all pick-up locations and their start-turn stops,

■ $E_R^{10} := \{(d_r, \delta_{\text{end}}) \in D \times \{\delta_{\text{end}}\} : r \in R\} \cup \{(\bar{d}_r, \delta_{\text{end}}) \in \bar{D} \times \{\delta_{\text{end}}\} : r \in R\}$, connecting all drop-off locations and their end-turn stops to the end depot.

Here, we write $v$ *precedes* $w$ to denote that the bus stations corresponding to $v$ precedes that corresponding to $w$ with respect to the corresponding line direction. We use $E_{\text{turn}} := E_R^4 \cup E_R^5$ to denote all the edges on which the vehicles turn. Each edge is only added once, even if it appears in multiple sets. The travel time of the edges is given by the original network, where a turn takes $t_{\text{turn}}$ and travel between two pick-ups or drop-offs at the same physical stop is instantaneous.

In our model, the binary variable $x_{i,j}^k$ denotes if a vehicle $k$ travels on defined arcs $(i,j) \in E_R$. Variables $z_k$ denote if vehicle $k$ is used in the solution.

We define that the vehicle loads $q_{\delta_{\text{start}}} := q_{\delta_{\text{end}}} = 0$, $q_i := 1$ for all $i \in P$ and $q_i := -1$ for all $i \in D$. Additionally, we set the service times $b_{\delta_{\text{start}}} := b_{\delta_{\text{end}}} = 0$ and $b_i = 0$ for all $i \in \bar{P} \cup \bar{D}$. Let $Q_i$ denote the passenger load of a vehicle departing a stop $i$ and let the continuous variable $B_i$ denote the start of service time at stop $i$. Note that these do not require an index for the vehicle $k$ as each node can be visited by at most one vehicle and the vehicles are homogeneous with a maximum capacity $Q_{\text{max}}$, as has been discussed in [7]. We require that $Q_{\delta_{\text{start}}}^k := Q_{\delta_{\text{end}}}^k = 0$ for all $k \in K$, thus removing these variables from the model.

To strengthen the model, we introduce a symmetry breaking constraint which enforces that vehicles of lower index are used first. Then, the full Location-Based model is given by:

$$\max_x \quad c_1 \left( \sum_{k \in K} \sum_{i \in P} t_{i,i+m} \cdot x_{i,i+m}^k - \sum_{k \in K} \sum_{\substack{j \in H_R: \\ (i,j) \in E_R}} \sum_{i \in H_R} t_{i,j} \cdot x_{i,j}^k \right) + c_2 \sum_{k \in K} \sum_{\substack{j \in H_R: \\ (i,j) \in E_R}} \sum_{i \in P} x_{i,j}^k$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{\substack{j \in H_R: \\ (i,j) \in E_R}} x_{i,j}^k \leq 1 \qquad \forall\, i \in P \tag{2a}$$

$$\sum_{\substack{j \in H_R: \\ (i,j) \in E_R}} x_{i,j}^k - \sum_{\substack{j \in H_R: \\ (m+i,j) \in E_R}} x_{m+i,j}^k = 0 \qquad \forall\, i \in P, k \in K \tag{2b}$$

$$\sum_{j \in P \cup \bar{P}} x_{\delta_{\text{start}},j}^k = 1 \qquad \forall\, k \in K \tag{2c}$$

$$\sum_{i \in D \cup \bar{D}} x_{i,\delta_{\text{end}}}^k = 1 \qquad \forall\, k \in K \tag{2d}$$

$$\sum_{\substack{j \in H_R: \\ (j,i) \in E_R}} x_{j,i}^k - \sum_{\substack{j \in H_R: \\ (i,j) \in E_R}} x_{i,j}^k = 0 \qquad \forall\, i \in N_R, k \in K \tag{2e}$$

$$1 - \sum_{j \in N_R} \sum_{\substack{i \in N_R: \\ (i,j) \in E_R}} x_{i,j}^k \leq M_3 \cdot x_{\delta_{\text{start}},\delta_{\text{end}}}^k \qquad \forall\, k \in K \tag{2f}$$

$$B_j \geq (B_{\delta_{\text{start}}}^k + b_{\delta_{\text{start}}} + t_{\delta_{\text{start}},j}) \cdot x_{\delta_{\text{start}},j}^k \qquad \forall\, j \in P \cup \bar{P} \cup \{\delta_{\text{end}}\}, k \in K \tag{2g}$$

$$B_{\delta_{\text{end}}}^k \geq (B_i + b_i + t_{i,\delta_{\text{end}}}) \cdot x_{i,\delta_{\text{end}}}^k \qquad \forall\, i \in D \cup \bar{D} \cup \{\delta_{\text{start}}\}, k \in K \tag{2h}$$

$$B_j \geq (B_i + b_i + t_{i,j}) \cdot \sum_{k \in K} x_{i,j}^k \qquad \forall\, i,j \in N_R : (i,j) \in E_R \tag{2i}$$

$$B_i \geq e_i + \sum_{j \in H_R \setminus \{i\}} \left( \max\{0, e_j - e_i + b_j + t_{j,i}\} \cdot \sum_{k \in K} x_{j,i}^k \right) \qquad \forall\, i \in N_R \tag{2j}$$

$$B_i \leq l_i - \sum_{j \in H_R \setminus \{i\}} \left( \max\{0, l_i - l_j + b_i + t_{i,j}\} \cdot \sum_{k \in K} x_{i,j}^k \right) \qquad \forall\, i \in N_R \tag{2k}$$

$$L_i = B_{i+m} - (B_i + b_i) \qquad \forall\, i \in P \tag{2l}$$

$$t_{i,i+m} \leq L_i \leq \alpha \cdot t_{i,i+m} \qquad \forall\, i \in P \tag{2m}$$

$$Q_j \geq q_j \cdot x_{\delta_{\text{start}},j}^k \qquad \forall\, j \in P \cup \bar{P}, k \in K \tag{2n}$$

$$Q_j \geq (Q_i + q_j) \cdot \sum_{k \in K} x_{i,j}^k \qquad \forall\, i,j \in N_R : (i,j) \in E_R \tag{2o}$$

$$0 \geq Q_i \cdot x_{i,\delta_{\text{end}}}^k \qquad \forall\, i \in D \cup \bar{D}, k \in K \tag{2p}$$

$$Q_i \leq Q_{\max} \cdot \left(1 - \sum_{k \in K} x_{i,j}^k\right) \qquad \forall\, (i,j) \in E_{\text{turn}} \tag{2q}$$

$$Q_i \geq -Q_{\max} \cdot \left(1 - \sum_{k \in K} x_{i,j}\right) \qquad \forall\, (i,j) \in E_{\text{turn}} \tag{2r}$$

$$Q_i \leq Q_{\max} \cdot \sum_{k \in K} \sum_{\substack{j \in H_R: \\ (i,j) \in E_R}} x_{i,j}^k \qquad \forall\, i \in H_R \tag{2s}$$

$$z_k \geq \frac{1}{|H_R|^2} \cdot \sum_{(i,j) \in E_R} x_{i,j}^k \qquad \forall\, k \in K \tag{2t}$$

$$z_k \geq z_{k+1} \qquad \forall\, k \in K \tag{2u}$$

Constraints (2a) and (2b) ensure each passenger is picked up at most once and is dropped-off by the same vehicle. Vehicles must start (2c) at $\delta_{\text{start}}$ and end at $\delta_{\text{end}}$ (2d) depots, maintaining flow conservation across all arcs (2e). Only unused vehicles may use the arc $(\delta_{\text{start}}, \delta_{\text{end}})$, as denoted by (2f), where we use a big-M constraint with $M_3 := |E_R|$. The service start times for leaving and entering the depot, as well as consistency across arcs, is handled by (2g)–(2i). Constraints (2j) and (2k) ensure time consistency regarding the requests time windows. The maximum ride time of each request is defined and bounded by constraints (2l) and (2m). Load constraints (2n)–(2s) ensure vehicles respect capacity limits at each bus stop as well as on turning arcs. Constraint (2t) counts the number of required vehicles. Finally, we use the symmetry breaking constraint (2u) to improve computational times.