The time-consistent dial-a-ride problem

Oscar Tellez, Samuel Vercraene Vercraene, Fabien Lehuédé, Olivier Péton, Thibaud Monteiro

The time consistent dial-a-ride problem

Tellez, Oscar[1] Vercraene, Samuel[1] Lehuédé, Fabien[2] Péton, Olivier[2]
Monteiro, Thibaud[1]

[1] Laboratoire DISP, INSA Lyon, Villeurbanne, France

[2] IMT Atlantique, LS2N, UMR CNRS 6004, Nantes, France

June 2021

Abstract

In the context of door-to-door transportation of people with disabilities, service quality considerations such as maximum ride time and service time consistency are critical requirements. To identify a good trade-off between these considerations and economic objectives, we define a new variant of the multi-period dial-a-ride problem called the time consistent dial-a-ride problem. A transportation planning is supposed to be time consistent if for each passenger, the same service time is used all along the planning horizon. However, considering the numerous variations in transportation demands over a week, designing consistent plan for all passengers can be too expensive. It is therefore necessary to find a compromise solution between costs and time consistency objectives. The time consistent dial-a-ride problem is solved using an epsilon-constraint approach to illustrate the trade-off between these two objectives. It computes an approximation of the Pareto front, using a matheuristic framework that combines a large neighbourhood search with the solution of set partitioning problems. This approach is benchmarked on time consistent vehicle routing problem literature instances. Experiments are also conducted in the context of door-to-door transportation for people with disabilities, using real data. These experiments support managerial insights regarding the inter-relatedness of costs and quality of service.

Keywords: Vehicle routing, Dial-a-ride problem, Healthcare logistics, Consistency, Set partitioning, Large neighborhood search.

1 Introduction

The design of efficient para-transit systems relies both on minimizing operational costs and on providing passengers with an adequate quality of service. In the operations research literature, the Dial-A-Ride Problem (DARP) is a well-known optimization problem that consists in designing minimal-cost vehicle routes to fulfill a set of transportation requests while satisfying a number of service quality requirements. Common applications concern door-to-door transportation of elderly people or people with disabilities. In Medico-Social Institutions (MSI) in France, transportation is considered to be the main expense after wages [1]. Transportation plans are defined on a yearly basis and partially revised several times a year whenever necessary. Due to the pressure to cut costs, this is often their main objective, although service quality criteria are also taken into account to define transportation plans.

The DARP formulation considers a single period, typically half a day. Passengers are generally subject to ride-time constraints: they must not be transported longer than a maximum predefined travel duration. In this paper, we examine the case of para-transit systems for people who need to be transported on a regular basis, for example handicapped workers or scholars. The DARP formulation is extended over multiple periods and each period has a known set of transportation requests from passengers.

We carried out a statistical study on field data which shows that 30% of passengers have a complete and regular schedule throughout the week. The other passengers may have a regular demand only a part of the week, or have slight variations in their planning (change of pickup or destination according to medical appointments, nights spent at the MSI, etc.).

Table 1 shows the percentage of users that have to be transported on five, four,..., one days of the week, respectively.

Number of demands per week	5	4	3	2	1
Percentage of passengers concerned	34.51%	9.33%	11.84%	15.42%	28.90%

Table 1: Frequency of transportation requests

For medical, cognitive or convenience reasons, it is desirable for passengers with regular demand to be proposed a regular service, i.e. to be picked up at the same time every day in the morning and dropped off at the same time on every evening. Each passenger's demand can have an impact on the schedule of other passengers on the same route. More particularly, the 28.9% of passengers with one weekly demand do not have consistency requirement, but these passengers "perturbate" the planning of all other passengers and make the consistency issue hard to solve.

During the week, the same fleet of vehicles must satisfy all transportation demands. To approach this objective, it is possible to have different routes and a different number of routes everyday. In the meanwhile, some practical constraints such as passenger time windows or maximum ride times must be respected and the overall transportation cost should be minimized.

This work extends the consistent vehicle routing problem of [9] to a consistent dial-a-ride problem. This allows to fully integrate real-life considerations such as time windows at pickup and delivery points, maximum ride times, and routes visiting several delivery points. The objective function to be minimized in [9] is the maximal number of *time classes* per passenger, where the time classes represents the number of significantly distinct schedules in the week. We reuse this definition of time classes and propose to lexicographically minimize the number of passengers with each number of time classes. In brief, we address a multi-period dial-a-ride problem and study the trade-off between service time consistency and transportation costs. As this problem introduces time consistency within a DARP setting, we call this new variant the time Consistent DARP (TC-DARP).

Section 2 presents how the TC-DARP is related to the existing literature in operations research. In particular, we discuss various definition of time consistency and motivate the use of a lexicographic objective function. In Section 3, we give a formal definition of the TC-DARP and formulate it as a mixed-integer linear program (MILP) with two objectives: minimizing transportation costs and lexicographically minimizing the number of time classes per passenger. Section 4 presents the solution method used to solve the TC-DARP. It a ε–constraint method, called SPεC, that repeatedly solves a set-partitioning (SP) formulation of the TC-DARP. In practice, this set partitioning formulation contains a only a small subset, called *pool of routes* of all possible routes. This pool of routes is generated by an auxiliary Large neighborhood Search (LNS) algorithm. When the current pool is not enough to ensure a good time consistency, new routes are appended to this pool. This requires solving an auxiliary optimization problem, which is a DARP with multiple time windows. The generation of these multiple time windows and the adaption of the LNS method used to solve it are presented in Section 5.

In section 6, we present computational results on consistent VRP instances and results on the case study that motivated this work. It concerns the transportation of people with disabilities in the area of Lyon, France. We collected data from the main carrier in the region[1], who works for multiple institutions and has a fleet of adapted vehicles. We show how using lexicographic minimization of the number of time classes offers a broader choice to decision makers to achieve a good trade-off between the transportation cost and the quality of service.

2 Related literature

The research presented in this paper is related to people transportation [6] and more particularly to the DARP, recently surveyed in [13, 23]. It also shares some similarities with the School Bus Routing Problem [25] or integrated vehicle scheduling problems arising in bus network design [3, 21], although research in this field seldom focus on modelling each passenger specific constraint. The research papers studying people transportation problems are generally focused on determining the best routes for a set of vehicles in order to serve the transportation requests of a single period. Some consistency aspects are studied in [4] by minimizing the deviation from predefined desired starting time for predefined trips. An alternative model was presented in [28] where the author minimize the deviation from ideal intervals between consecutive service trips of the same link. The consistency aspects in these papers aim at evenly spreading service times at a bus stop over the horizon. Note that we do not allow transfers between

[1] Synergihp Rhône-Alpes: `www.synergihp-ra.fr`

vehicles as it is generally done in public transit systems. The mono-period DARP of our application has been presented in [37]. In this paper, we focus on the consistency aspects that appear in the multi-period version of the problem.

The integration of time consistency appeared recently in the vehicle routing problem (VRP) literature. Applications were first identified in the context of fast parcel delivery [11] and were rapidly extended to passenger transportation [9]. Readers interested in an extensive review on vehicle routing with consistency considerations can refer to Kovacs et al. [16]. Consistency in vehicle routing problems can be divided into three main categories: service time consistency, driver consistency, and territory consistency. *Service time consistency* means that regular customers are scheduled to be served at approximately the same time in the planning horizon. As the main focus of our paper, the service time consistency will be detailed in the next section.

Driver consistency consists in minimizing the number of different drivers assigned to each passenger during the planning horizon. The aim is to reinforce the relationship between drivers and passengers in order to improve the quality of service. Braekers and Kovacs [2] computed the average cost of a solution where each passenger was served by one, two and three drivers, respectively, showing that a solution with two drivers can be near optimal whereas solutions with one driver are 10% costlier on average. Other approaches using soft constraints have yielded similar conclusions [30, 22]. In Feillet et al. [9], drivers are assigned to routes *a posteriori*, so that service time consistency and driver consistency are considered as independent problems in a lexicographical way.

Territory consistency aims at increasing drivers efficiency through their knowledge of the geographical area in which they operate. A common way of addressing territory consistency is to design independent districts in advance, where independent routing problems are solved every day. This approach was studied in [18, 40, 30, 29].

This paper focuses on service time consistency applied to a Dial-a-Ride Problem (DARP). In contrast to the VRP, the DARP considers one origin and destination for each user and a maximum ride time. The main applications of the DARP concern door-to-door transportation of people, particularly elderly or disabled people [13, 38, 17].

2.1 Service time consistency models

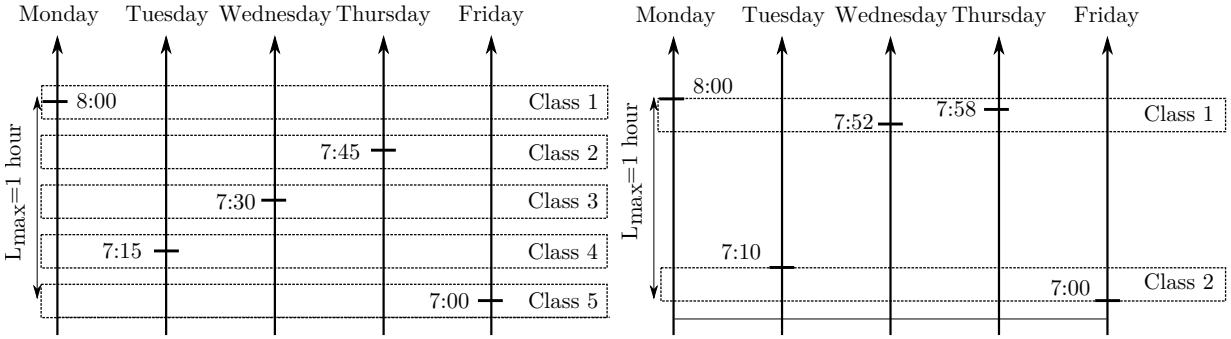
Service time consistency consists in serving regular needs at approximately the same hour throughout the whole planning horizon. This is modeled either by hard constraints, that is, imposing an acceptable level of service time variation, or by soft constraints, that is, penalizing service time variations in the objective function.

Groër et al. [11] defined the maximum arrival time variation as the difference between the latest and earliest service times throughout the whole planning horizon, for each customer. This consistent VRP (conVRP) is an extension of the multi-period VRP where the maximal arrival time variation is bounded above by a constant value $L_{\max}$. However, this measure, initially proposed for the small package shipping industry, has some practical drawbacks in the context of passenger transportation.

Feillet et al. [9] define a passenger-oriented time consistency model based on the concept of time classes. They assume that very small variations (e.g. ± 5 minutes) in service time are not significant for passengers, especially considering approximations and variations due to traffic conditions or unexpected events. Passengers are sensitive to the number of significantly different service times proposed in a week. Similar times are regrouped in the same time class. Consistency is then improved by minimizing the maximum number $C_{\max}$ of time classes over all passengers. The difference between this measure and $L_{\max}$ is highlighted in Figure 1.

Figures 1(a) and 1(b) represent the service time of a passenger in two distinct solutions. Each vertical line represents the service time from Monday to Friday. In Figure 1(a), as service times are evenly spread between 7:00 and 8:00, there are 5 time classes. In Figure 1(b), these times can be grouped into two intervals of 10 minutes: [7:00-7:10] and [7:50-8:00]. This passenger is said to have 2 time classes. While both solutions have the same value $L_{\max} = 1$ hour, they do not offer the same consistency to passengers as far as service time is concerned. In our application, a measure based on the number of time classes offers a better quality of service than a solution measure with $L_{\max}$.

There is however one limitation related to objective $C_{\max}$. It is the largest number of time classes in the solution, for all passengers. Hence, a solution with 99% of passengers who have $C_{\max}$ time classes is equivalent to another solution where only 1% of passengers are the same situation. In order to overcome this limitation, we propose a lexicographic optimization. We first minimize the number of passengers from the highest to the lowest number of time classes: first passengers with $C_{\max}$ time classes, then



(a) Solution with $C_{\max} = 5$ time classes

(b) Solution with $C_{\max} = 2$ time classes

Figure 1: Comparison, for one passenger, of two solutions having the same $L_{\max}$ but different numbers of time classes. The width of a time class is 10 minutes.

$C_{\max} - 1$, $C_{\max} - 2$ and so forth.

To the best of our knowledge, this approach is a new refinement of the Feillet et al. [9] model. Still, according to the passenger transportation company, many good trade-off solutions can be found between the cost optimal solution with $C_{\max} = 2$ and the cost optimal solution with $C_{\max} = 1$. According to the fair optimization literature [24], the proposed model corresponds to a lexicographic minimax refinement of the min-max model, using counting functions. We show that this lexicographic objective is easily adapted to the context of passenger transportation.

2.2 Solution approaches for time consistent routing problems

In the conVRP model introduced by Groër et al. [11], the objective is to optimize service time consistency ($L_{\max}$) without compromising a perfect driver consistency (1 driver per customer). These authors proposed a record-to-record travel algorithm and developed benchmark instances for up to 100 customers.

The consistency measure $L_{\max}$ has been used in several subsequent papers (i.e. [34, 35, 14, 15, 19, 39]). The current best results on the Groër et al. [11] benchmark instance set was obtained by Xu and Cai [39], who proposed a Variable Neighborhood Search procedure using dedicated local search methods for quickly finding local optima. This approach is based on improving template solutions generated by 3 different shaking methods. A problem extension, denoted the genConVRP, is proposed by Kovacs et al. [15] in which: routes do not necessarily start at the same time, customers are associated with AM/PM time windows, and a maximum number of drivers per customer is defined. Subramanyam and Gounaris [33] propose a branch-and-cut framework to solve the consistent traveling salesman problem which is a particular case of the conVRP using a single vehicle without capacity constraints. They solve instances, randomly generated, with up to 51 customers.

The time Consistent VRP (TCVRP) of Feillet et al. [9] is solved with a dedicated Large Neighborhood Search (LNS) framework. At each iteration, the routes of all periods are destroyed. A VRP with multiple time windows and no waiting time (VRPmTW-nw) is defined in order to decrease the number of time classes of one passenger. A branch-and-price heuristic is used to recreate the routes. The minimum cost solutions for $C_{\max} = 1$ to 5 are saved in the process.

Another related problem is the Time Window Assignment Vehicle Routing Problem (TWAVRP) introduced by Spliet and Gabor [32]. In the TWAVRP, a single time window of fixed width has to be assigned to some regular customers before the effective daily demand is known. The assignment is based on a set of demand scenarios, each of which is associated with a given probability. The objective is to minimize the expected transportation cost. The TWAVRP is a particular case of the genConVRP if scenarios are seen as periods and the number of drivers per customer is set to infinity. However, the objective differs: genConVRP optimizes the total transportation cost and TWAVRP the average transportation cost. A branch-price-and-cut algorithm is proposed to optimally solve instances with up to 25 customers. Spliet and Desaulniers [31] propose a variant, called the discrete time window assignment vehicle routing problem, where the chosen time windows are selected from a discrete set. Considering its formulation, the TWAVRP corresponds to the case where each user should have exactly one time class in the TCVRP, with each day in the TCVRP corresponding to a scenario with probability one in the TWARVP.

Consistency issues are also often related to having stochastic customers in the VRP [27]. For example,

Sungur et al. [34] use a combination of robust optimization in a first phase master problem, and stochastic programming with recourse to daily schedules to address the uncertainty in service times and customer occurrence. Erera et al. [8] investigate the opportunity to give a main fixed route as well as a backup one to frequent customers in a stochastic context.

Finally, the question of service time consistency presents some similarities with some non-periodic applications such as the synchronization of multiple vehicles at the same node. In this case, the arrival time of multiple vehicles at a given location should be synchronized in order to perform a collective operation. The vehicles then continue their routes independently. A survey on synchronization in VRP is given by Drexl [7].

2.3 Contributions with respect to the literature

The literature review shows that there is still some gap between what has been proposed in the literature and a practical implementation of time consistency for a DARP application. In this paper, we reuse the notion of time classes introduced in Feillet et al. [9] and extend this previous work from the time consistent VRP to the time consistent DARP. In particular, we consider time windows both at pickup and delivery locations and maximum ride times for all passengers. Another contribution is to better explore solutions where some passengers accept several time classes. To aggregate of the number of time classes over all passengers, we refine the $C_{\max}$ minimization approach of [9]. Our lexicographic approach minimizes the number of passengers with the highest to the lowest number of time classes: first passengers with $C_{\max}$ time classes, then with $C_{\max} - 1$, $C_{\max} - 2$ and so forth. As a result, the proposed method finds a more complete Pareto front, showing more trade-off between cost and consistency.

Due to the time windows at delivery locations, routes might contain waiting times. Similarly to Kovacs et al. [15], we consider that the departure time of routes can be changed in order to improve consistency. To achieve these objectives, we revisit the method of [9], in particular by integrating the LNS matheuristic of [37] in the method to design the routes of each week day. As in [9], our algorithm includes a component that recreates the routes of one day, taking account of the times at which users are served on the other days. For this component, a specific case of the DARP with multiple Time Windows is defined. It is solved by adapting the previous LNS algorithm through a new route scheduling algorithm that integrates the constraints and objectives of our problem. The generated routes are combined in an ε-constraint method, with routes selection strategies, to exhibit the TC-DARP Pareto front.

3 Modeling the time consistent dial-a-ride-problem (TC-DARP)

In this section, we present the TC-DARP problem settings and give its MILP formulation. Tables 2, 3 and 4 synthesize the mathematical notations for the sets, data and variables, respectively.

Users

In the remainder of this paper, we refer to the passengers of the transportation system as *users*. The set of users is denoted by $\mathcal{U}$. Each user $u \in \mathcal{U}$ is carried between a pickup node $p_u \in \mathcal{P}$ and a delivery node $d_u \in \mathcal{D}$, where $\mathcal{P}$ and $\mathcal{D}$, denote the set of all pickup and delivery nodes, respectively. Moreover, each user has a maximum ride time $\overline{T}_u$, and occupies a particular space $v \in \mathcal{J}$ in vehicles.

Given a planning horizon $\mathcal{T}$ (typically one week), each user can be service at most once in any period, the user's demand is modeled by a vector of binary values of size $|\mathcal{T}|$. More precisely, we define $\beta_u^t \in \{0, 1\}$ for all $u \in U, t \in \mathcal{T}$, where $\beta_u^t = 1$ if user u requires transportation at period t and 0 otherwise. Note that if several users have a common origin or destination, nodes are duplicated so that each pickup node and each delivery node corresponds to exactly one user. Similarly, we only consider users with stable demand within the planning horizon (same pickup, same delivery). Actually, users with several addresses do not expect time consistency. Thus, they are considered as distinct users with separate plannings.

Vehicles

We assume a homogeneous and unlimited fleet of vehicles based at a single depot o. In practice, vehicles are equipped with several spaces (e.g. seats, spaces for wheelchairs) that can receive different categories of users. We thus consider vehicles' capacity as a vector indicating the number of available places in each space. It is denoted by $\mathcal{Q} = \{Q_1, \ldots, Q_{|\mathcal{J}|}\}$, where $\mathcal{J}$ represents the set of passenger spaces (seats, wheelchairs, etc.)

Graph representation

The TC-DARP is defined on a directed graph which node set is $\{\mathcal{P} \cup \mathcal{D} \cup o\}$ and the arcs set containing the following arcs: (o, i) where $i \in \mathcal{P}$; (i, j) where $i, j \in \mathcal{P} \cup \mathcal{D}, i \neq j$; and (i, o) where $i \in \mathcal{D}$. Each node i is associated with a service duration ζ_i and a time window $[a_i, b_i]$. Every arc (i, j) represents the fastest path from node i to node j and is associated with a travel time t_{ij} and a distance d_{ij}.

Routes

We propose a route-based MILP formulation of the TC-DARP. The set of all routes is denoted by Ω. A route $\omega \in \Omega$ is any feasible sequence of nodes visited by the same vehicle. More precisely, a route is feasible if it starts and finishes at the depot o, serves a set of users by visiting their pickup node and their delivery node (in this order), while respecting time windows at all nodes visited, users ride time constraints and vehicle capacity constraints. Given a potentially large set of feasible routes, the MILP model aims at selecting a subset of routes that cover the whole demand. The binary decision variable y^t_ω is equal to 1 if route $\omega \in \Omega$ is selected at period $t \in \mathcal{T}$.

Each selected route is associated with one vehicle. For each route, we consider three types of costs: a fixed cost λ paid if the route is used at least once in the planning horizon, a cost per kilometer τ related the fuel consumption, and a cost per hour α related to the driver's cost. For a given route ω which length and duration can be easily calculated, the sum of fuel and driver's related cost is denoted by π_ω.

Routes scheduling

The pickup and delivery nodes $i \in \mathcal{P} \cup \mathcal{D}$ on a route ω are associated with a time $H_{i,\omega}$. $H_{i,\omega}$ is the earliest possible service time of node i in a schedule of ω that minimizes its duration, satisfies time windows at each node and users maximum ride time constraints. In order to improve time consistency, the departure of a route can be postponed provided the route remains feasible. The maximal possible shift for a route $\omega \in \Omega$ is denoted by Δ^+_ω. The shift of route $\omega \in \Omega$ departure time at period $t \in \mathcal{T}$ is expressed by continuous variables $\delta^t_\omega \in [0, \Delta^+_\omega]$. Adding waiting time to improve consistency is possible only at the depot. More precisely, once vehicles have left the depot, waiting times are introduced only when the vehicle arrives at some pickup or delivery point before the opening of time windows. Other waiting times are not authorized, even though they would sometimes improve consistency. This rule is realistic with respect to drivers practice. Indeed, a driver would not wait at a pickup location within the user's time windows, possibly with other users on board, just to improve its consistency. Appendix A.2 details the scheduling procedure for the calculation of Δ^+_ω and the satisfaction of the time windows and ride-time constraints.

Time classes

The set of selected routes determines the service times for all users. Given these service times, it is possible to assign each visit to a particular user to time classes. The binary variable z^t_{uc} is equal to 1 if user $u \in \mathcal{U}$ is assigned to time class $c \in \mathcal{C}$ at period $t \in \mathcal{T}$, and 0 otherwise. For example, in Figure 1(b), Tuesdays and Fridays belong to class 1. Thus, assuming that the corresponding user is denoted $\bar{u}$, we have $z^2_{\bar{u}1} = z^5_{\bar{u}1} = 1$. Similarly, Mondays, Wednesdays and Thursdays belong to class 2, then $z^1_{\bar{u}2} = z^3_{\bar{u}2} = z^4_{\bar{u}2} = 1$.

Binary variables μ_{uc} indicate which time classes from set $\mathcal{C}$ are actually used by user u. In the preceding example, the user $\bar{u}$ has two classes, thus $\mu_{\bar{u}1} = \mu_{\bar{u}2} = 1$. Without loss of generality, we assume that if a user has $\bar{c}$ time classes ($1 \leq \bar{c} \leq 5$), these time classes are sequentially numbered from 1 to $\bar{c}$.

$\mathcal{U}$	set of users
$\mathcal{T}$	set of time periods
$\mathcal{T}_u$	set of time periods in which user $u \in \mathcal{U}$ needs to be transported
$\mathcal{J}$	set of user spaces in vehicles
Ω	set of all routes
Ω_u	set of routes serving user $u \in \mathcal{U}$
$\mathcal{C}$	set of time classes

Table 2: Sets

β_u^t	equal to 1 if user $u \in \mathcal{U}$ must be serviced in period $t \in \mathcal{T}$, and 0 otherwise
π_ω	cost of route $\omega \in \Omega$
λ	weekly vehicle fixed cost
$H_{u\omega}$	earliest time service to user $u \in \mathcal{U}$ by route $\omega \in \Omega$
Δ_ω^+	maximum time shift of route $\omega \in \Omega$
Λ	time class width

Table 3: Data

Binary Variables	
$y_\omega^t \in \{0,1\}$	=1 if route $\omega \in \Omega$ is selected at period $t \in \mathcal{T}$, and 0 otherwise
$z_{uc}^t \in \{0,1\}$	=1 if user $u \in \mathcal{U}$ is assigned to time class $c \in \mathcal{C}$ at period $t \in \mathcal{T}$, and 0 otherwise
$\mu_{uc} \in \{0,1\}$	=1 if user $u \in \mathcal{U}$ uses time class $c \in \mathcal{C}$, and 0 otherwise
Other variables	
$\delta_\omega^t \in [0, \Delta_\omega^+]$	time shift (used margin) of route $\omega \in \Omega$ at period $t \in \mathcal{T}$
$s_{uc}^-, s_{uc}^+ \in \mathbb{R}^+$	lower and upper bounds for the time class $c \in \mathcal{C}_u$ for user $u \in \mathcal{U}$
$h_u^t \in \mathbb{R}^+$	beginning of service for user $u \in \mathcal{U}$ at period $t \in \mathcal{T}$
ν	number of vehicles needed for the whole planning horizon
m_c	number of users having $c \in \mathcal{C}$ time classes (post-processed variable)

Table 4: Summary of variables

Objective functions

The TC-DARP is a bi-objective problem which consists in selecting a subset of routes from Ω such that transportation requests on the planning horizon are satisfied within their time windows and maximum ride times.

The first objective is the minimization of transportation costs, that include the fleet fixed cost $\lambda\nu$ and the cost of all routes used in the solution.

$$\min f = \lambda\nu + \sum_{\omega\in\Omega}\sum_{t\in\mathcal{T}} \pi_\omega y_\omega^t. \tag{1}$$

The second objective minimizes time inconsistency, which is modeled with a lexicographical refinement of the time class model proposed by Feillet et al. [9]. The expression used in the MILP model is the following:

$$\text{lexmin}\, \hat{\boldsymbol{g}} = \left(\sum_{u\in\mathcal{U}} \mu_{u|\mathcal{C}|}, \ldots, \sum_{u\in\mathcal{U}} \mu_{u2} \right). \tag{2}$$

This expression lexicographically minimizes the number of people having more than c time classes, where c decreases from $|\mathcal{C}|$ to 2. The expression $\sum_{u\in\mathcal{U}} \mu_{uc}$ counts the number of users with c or more time classes. This is equivalent to the lexicographical minimization of the number of users whose number of time classes is exactly $|\mathcal{C}|$, $|\mathcal{C}|-1$, down to 1, respectively. In the remainder of the paper, we denote by

$$\text{lexmin}\, \boldsymbol{g} = \left(m_{|\mathcal{C}|}, \ldots, m_1\right) \tag{3}$$

the alternative formulation of this objective, where m_c denotes the number of users having c time classes. It is post-processed from the values of the μ_{uc} variables using the following expressions:

$$\begin{cases} m_1 = |\mathcal{U}| - \sum_{u\in\mathcal{U}} \mu_{u2} \\ m_c = \sum_{u\in\mathcal{U}} \mu_{uc} - \sum_{u\in\mathcal{U}} \mu_{u,c+1} \quad \forall c \in \{1, \ldots, |\mathcal{C}|-1\} \\ m_{|\mathcal{C}|} = \sum_{u\in\mathcal{U}} \mu_{u|\mathcal{C}|} \end{cases} \tag{4}$$

Constraints

The set of TC-DARP feasible solutions is defined by the following constraints:

$$\sum_{\omega \in \Omega_u} y_\omega^t = \beta_u^t \quad \forall u \in \mathcal{U}, t \in \mathcal{T}, \tag{5}$$

$$\sum_{\omega \in \Omega} y_\omega^t \leq \nu \quad \forall t \in \mathcal{T}, \tag{6}$$

$$\sum_{c \in \mathcal{C}} z_{uc}^t = 1 \quad \forall u \in \mathcal{U}, t \in \mathcal{T}_u, \tag{7}$$

$$h_u^t = \sum_{\omega \in \Omega_u} (H_{u\omega}\, y_\omega^t + \delta_\omega^t) \quad \forall u \in \mathcal{U}, t \in \mathcal{T}_u, \tag{8}$$

$$\delta_\omega^t \leq \Delta_\omega^+ y_\omega^t \quad \forall \omega \in \Omega, t \in \mathcal{T}, \tag{9}$$

$$s_{uc}^- \leq h_u^t + M(1 - z_{uc}^t) \quad \forall c \in \mathcal{C}, u \in \mathcal{U}, t \in \mathcal{T}_u, \tag{10}$$

$$h_u^t \leq s_{uc}^+ + M(1 - z_{uc}^t) \quad \forall c \in \mathcal{C}, u \in \mathcal{U}, t \in \mathcal{T}_u, \tag{11}$$

$$s_{uc}^+ - s_{uc}^- = \Lambda \quad \forall c \in \mathcal{C}, u \in \mathcal{U}, \tag{12}$$

$$s_{uc}^+ \leq s_{u,c+1}^- \quad \forall c \in \mathcal{C}/\{|\mathcal{C}|\}, u \in \mathcal{U}, \tag{13}$$

$$\sum_{\omega \in \Omega_u} y_\omega^t = \sum_{c \in \mathcal{C}} z_{uc}^t \quad \forall u \in \mathcal{U}, t \in \mathcal{T}_u, \tag{14}$$

$$z_{uc}^t \leq \mu_{uc} \quad \forall c \in \mathcal{C}, u \in \mathcal{U}, t \in \mathcal{T}_u, \tag{15}$$

$$\mu_{u,c+1} \leq \mu_{uc} \quad \forall c \in \mathcal{C}, \forall u \in \mathcal{U}, \forall t \in \mathcal{T}, \tag{16}$$

$$y_\omega^t, z_{uc}^t, \mu_{uc} \in \{0,1\} \quad \forall c \in \mathcal{C}, u \in \mathcal{U}, t \in \mathcal{T}, \omega \in \Omega, \tag{17}$$

$$\delta_\omega^t, h_u^t s_{uc}^-, s_{uc}^+, \nu \in \mathbb{R}^+ \quad \forall c \in \mathcal{C}, u \in \mathcal{U}, t \in \mathcal{T}, \omega \in \Omega. \tag{18}$$

Constraints (5) are partitioning constraints ensuring the satisfaction of the users demand. Constraints (6) count the number of vehicles needed during the planning horizon. Constraints (7) state that each user served in period $t \in \mathcal{T}$ should be given a single time class. Constraints (8) determine the service time for each user of route ω when its departure is shifted by the value δ_ω^t. Constraints (9) state that if a route ω is selected in period t, then its time shift δ_ω^t should not exceed the value Δ_ω^+.

Constraints (10) and (11) linearize the following logical expression:

$$z_{cu}^t = 1 \Rightarrow s_{uc}^- \leq h_u^t \leq s_{uc}^+ \quad \forall u \in \mathcal{U}, t \in \mathcal{T}_u, c \in \mathcal{C}. \tag{19}$$

They state that if a user u is assigned to the time class c at period t, then its service time should be within the bounds of this time class c. Constraints (12) set the width of a time class. Constraints (13) avoid overlap between time classes. Constraints (14) link the number of routes that serve one given user and the number of time class variables. Constraints (15) define variables μ_{uc} necessary for counting the number of time classes of each user. Constraints (16) ensure that time classes are defined in increasing order. For example, time class #2 is allocated to a user only if time class #1 already exists and is not compatible with a given service time. Finally, the definition of variables is given by constraints (17) and (18).

4 Solution method

This section presents the Set Partitioning-based ε–constraint matheuristic, denoted SPεC, that has been designed to solve the bi-objective TC-DARP. This method iteratively solves Set Partitioning Problems (SPPs) in an ε-constraint framework. SPPs correspond to a route-based formulation of the TC-DARP considering a subset of the whole set of feasible routes. This subset of routes, called *pool of routes*, contains routes generated by an auxiliary heuristic solution method to solve the TC-DARP. Here we generate the pool of routes by using a Large Neighborhood Search algorithm (LNS). This section is structured as follows: Section 4.1 presents the general framework of SPεC, that traces a Pareto front approximation between the two objectives of the TC-DARP. Section 4.2 details the optimization procedure for solving the TC-DARP with a single objective. Next subsections are focus on two more specific topics: the initialization of the SPεCframework (Section 4.3), the selection of the pool of routes from a larger set of routes (Section 4.4).

4.1 The SPεC matheuristic framework

The general framework, introduced in Algorithm 1, is based on an ε-constraint procedure [12, 5]. It finds an approximation of the Pareto front between the two objectives of the TC-DARP: the transportation cost f, and the time inconsistency $\boldsymbol{g}$. This is illustrated in Figure 2.

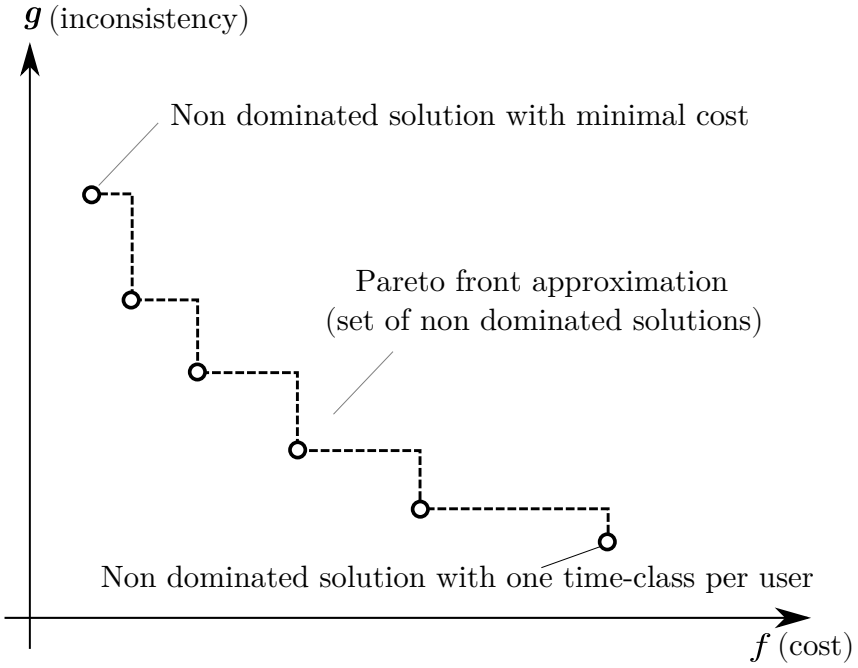


Figure 2: Pareto front approximation designed by the SPεC algorithm

In a nutshell, the algorithm starts from the best solution found by lexicographically minimizing cost and then inconsistency. Then, the inconsistency objective is progressively improved by allowing an increase of the transportation cost by ε percent. Every time a new non-dominated solution is found, it is stored in the Pareto front approximation. The procedure stops when every user has 1 time class.

Algorithm 1 presents the SPεC framework, the ε–constraint matheuristic. In this algorithm, a pool of routes $\mathcal{L}$ is generated together with the initial solution. Two types of solution are used, a temporary solution S (line 3) and a best found solution S^*. They are initialized with the procedure described in Section 4.3 (line 4). This procedure solves a multi-period DARP in which cost f is minimized. The routes found while solving this multi-period DARP are appended to the pool $\mathcal{L}$. The cost of solution S^* is taken as the cost upper limit $\bar{f}$ (line 5).

Lines 7 to 15 describe an iteration of the algorithm. The procedures described in lines 7 and 8 aim at finding a new temporary solution S, as detailed in Figure 3. First inconsistency $\boldsymbol{g}$ is minimized subject to a maximal cost constraint. Given that this procedure starts with a feasible solution S^*, it results in a solution S such that $\boldsymbol{g}(S) \leq_{lex} \boldsymbol{g}(S^*)$. Then the cost objective f is minimized subject to a maximal inconsistency level $\boldsymbol{g}(S)$. During these two procedures, pool $\mathcal{L}$ is updated with new routes.

If the temporary solution S is strictly better than S^* for at least one of the objectives (i.e. if $f(S) < f(S^*)$ or $\boldsymbol{g}(S) <_{lex} \boldsymbol{g}(S^*)$, line 9) then solution S^* is updated with S (line 10) and S^* is added to the Pareto front approximation (line 11). Otherwise, the value of ε is geometrically increased by a factor ϕ (line 13). At the end of an iteration, the cost limit $\bar{f}$ is updated based on the cost of S^* (line 15) and ε value.

Note that the size of pool $\mathcal{L}$ increases at each iteration, which can eventually cause memory issues. Thus, the routes in $\mathcal{L}$ are ordered in non-decreasing order of the consistency measure among all TC-DARP solutions found so far that contains these routes, and the first N_{max} routes are kept (line 16).

Since the fleet size is not limited, there is an extreme point in the Pareto front such that every user has only one time class (i.e. $\boldsymbol{g} = (0, \ldots, |\mathcal{U}|)$). Therefore the stopping criterion used in line 6 is met when $\boldsymbol{g} = (0, \ldots, |\mathcal{U}|)$.

Algorithm 1: The SPεC framework

Parameters ε : initial value of epsilon, ϕ : increase factor of epsilon,
Result: Pareto front approximation

`/* Initialization */`

1 $ParetoFront \leftarrow \emptyset$
2 $\mathcal{L} \leftarrow \emptyset$: Pool of routes
3 $S \leftarrow \emptyset$: temporary solution
4 $(S^*, \mathcal{L}) \leftarrow initialize()$ `/* See Section 4.3 */`
5 $\bar{f} \leftarrow f(S^*)$: cost upper limit

`/* Iterations */`

6 **while** *stopping criterion is not met* **do**
  `/* Optimize inconsistency and cost objectives, see Section 4.2 */`
7 $(S, \mathcal{L}) \leftarrow solveMonoTCDARP(\text{lexmin}\, \boldsymbol{g}, f \leq \bar{f}, S^*, \mathcal{L})$
8 $(S, \mathcal{L}) \leftarrow solveMonoTCDARP(\min f, \boldsymbol{g} \leq_{lex} \boldsymbol{g}(S), S, \mathcal{L})$
  `/* Update solution */`
9 **if** $(f(S) < f(S^*)) \vee (\boldsymbol{g}(S) <_{lex} \boldsymbol{g}(S^*))$ **then**
10 $S^* \leftarrow S$
11 Update $ParetoFront$ with solution S^*
12 **else**
13 $\varepsilon \leftarrow \phi \times \varepsilon$
14 **end**
  `/* End of one iteration */`
15 Update epsilon constraint: $\bar{f} \leftarrow f(S^*) \times (1 + \varepsilon)$
16 Limit the size the $\mathcal{L}$ to N_{max}
17 **end**
18 **return** $ParetoFront$

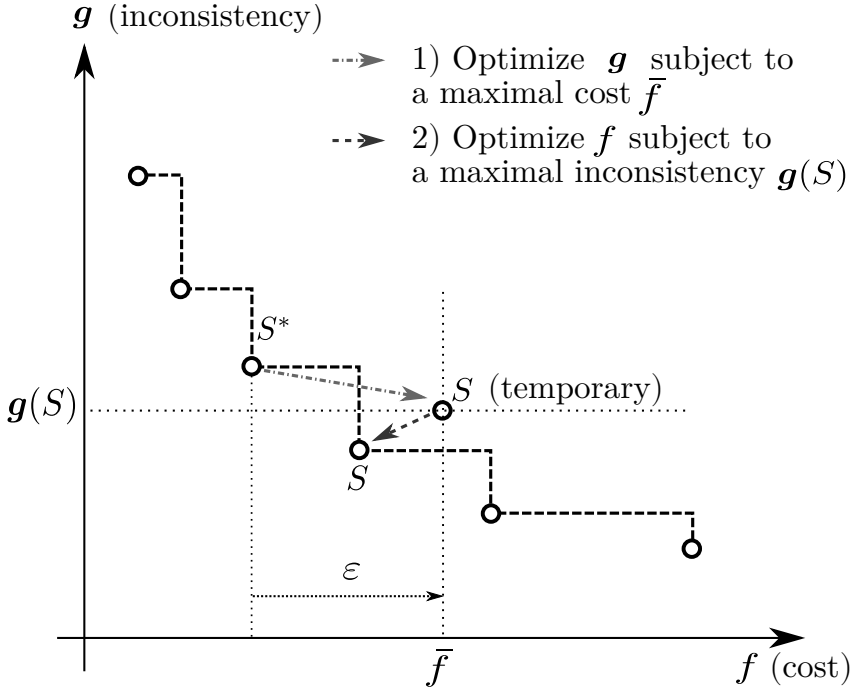


Figure 3: Pareto front exploration with the two optimization procedures of one iteration (Algorithm 1 lines 7 and 8).

4.2 Mono-objective optimization procedure

Line 7 of Algorithm 1 concerns the minimization of inconsistency subject to a maximal cost constraint. Line 8 concerns the minimization of cost subject to a maximal inconsistency constraint. These two consistent DARP formulations with a single objective are solved with Algorithm 2. The arguments of Algorithm 2 are the objective function z to be minimized (z is a generic notation representing either f or $\boldsymbol{g}$), an upper bound $\mathcal{E}$ on the other objective, a solution S_0 and a pool of routes $\mathcal{L}$.

Algorithm 2: solveMonoTCDARP(z, $\mathcal{E}$, S_0, $\mathcal{L}$)

Arguments: z: objective, $\mathcal{E}$: epsilon constraint, S_0: solution, $\mathcal{L}$: pool of routes.
Parameters: $MaxIter$: maximum number of iterations without improvement, $t_{\max}$: solver time limit, N : number of routes selected at each iteration of a TC-DARP problem
Result: best solution found S^*

1 $S \leftarrow \emptyset$: current solution
2 $S^* \leftarrow S_0$: best solution found
3 $\mathcal{L}' \leftarrow \emptyset$: restricted pool of routes
4 $itNonImp \leftarrow 0$: number of iterations without improving S^*
5 $l_{memory} \leftarrow \emptyset$: initialize memory of selected routes
6 $genNewRoutes \leftarrow false$
7 **while** $itNonImp < MaxIter$ **do**
  /* 1. Select routes */
8 **if** $genNewRoutes = false$ **then**
9 $l \leftarrow selectRoutes(\mathcal{L}, N, l_{memory})$ /* See Section 4.4 */
10 **else**
11 $\mathcal{L}_{new} \leftarrow generateNewRoutes(S^*)$ /* See Section 5 */
12 $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{L}_{new}$: Save new routes
13 $l \leftarrow selectRoutes(\mathcal{L}_{new}, N, l_{memory})$ /* See Section 4.4 */
14 **end**
15 $\mathcal{L}' \leftarrow \mathcal{L}' \cup l$
16 $l_{memory} \leftarrow l_{memory} \cup\ l$
  /* 2. Solve a TC-DARP on subset l */
17 Compute S by solving the TC-DARP problem with objective z and constraints (5)–(18), epsilon constraint $\mathcal{E}$, with $\Omega = \mathcal{L}'$, time limit $t_{\max}$, and warm start on S^*
  /* 3. Pool management */
18 **if** *TC-DARP is not solved to proven optimality* **then**
19 $\mathcal{L}' \leftarrow \emptyset$
  /* 4. Update best solution and $itNonImp$ */
20 **if** $z(S) < z(S^*)$ **then**
21 $S^* \leftarrow S$
22 $itNonImp \leftarrow 0$
23 **else**
24 $itNonImp \leftarrow itNonImp + 1$
25 $genNewRoutes \leftarrow \neg\ genNewRoutes$
26 **end**
27 **end**
28 **return** $S^*, \mathcal{L}$

In lines 1 to 5, the main variables are initialized. The current solution S, a restricted pool of routes $\mathcal{L}' \subset \mathcal{L}$ and a list of selected routes are initialized as empty sets. The best solution found S^* is initialized to S_0. The variable $itNonImp$, initialized with the value 0, counts the number of iterations with no improvement of the objective function. Finally, the boolean variable $genNewRoutes$ indicates if the pool $\mathcal{L}$ must be enriched with new routes. It is initialized with the value FALSE.

Each iteration of Algorithm 2 consists of four steps: 1) selecting a subset of routes from $\mathcal{L}$; 2) solving the mono-objective TC-DARP given the set of selected routes; 3) managing the pool of routes; and 4) updating the best solution. This process iterates until $MaxIter$ iterations without any improvement of S^* having been performed.

In Step 1, a subset l of N routes is selected from a larger pool of routes, which is either $\mathcal{L}$ (line 9) or a new pool $\mathcal{L}_{new}$ determined from the set of service times in solution S^* (lines 11-13). The process followed to generate these new routes is detailed in Section 5. New routes in $\mathcal{L}_{new}$ are saved in $\mathcal{L}$ (line 12) to enable their selection in further iterations. The procedure for selecting the N routes is the same regardless of the pool. It is called *selectRoutes* and it is detailed in Section 4.3. In this procedure, routes are selected based on their performance and whether they were chosen on a previous iteration. Selected routes are then saved in memory at every iteration (line 16).

Step 2 consists in solving a TC-DARP defined by the objective function z and constraints (5–18), with epsilon constraint $\mathcal{E}$, where the set of routes Ω is the restricted pool $\mathcal{L}'$. This model is solved with a time limit set at $t_{\max}$ and a warm start on S^*. Depending whether z represents f or $\boldsymbol{g}$, the objective z to be minimized is either the cost or the inconsistency. In Step 3 (lines 18-19), the restricted pool of routes is cleared if the TC-DARP at Step 2 could not be solved to proven optimality. This mechanism, inspired by previous research [10, 37], keeps the pool size manageable.

Step 4 updates the current best solution S^* and the counter $itNonImp$ of iterations without improvement. Thanks to the warm start, the value of the objective function $z(S)$ cannot increase from one iteration to another. Given this property, when the value of $z(S^*)$ has not been improved for $MaxIter$ iterations, we suppose that the algorithm has reached a local optimum. The parameter $itNonImp$ is used as a stopping criterion. When the solution is not improved, the value of the boolean $genNewRoutes$ is modified in order to diversify the process by switching between pools $\mathcal{L}$ and $\mathcal{L}_{new}$ in step 1.

4.3 Initialization

The initial solution of Algorithm 1 can be obtained by executing a variant of Algorithm 2 ignoring all consistency considerations. This amounts to solve a separate dial-a-ride problem for each time period and the corresponding mathematical model, called MP-DARP (multi-period DARP) is as follows:

$$\min f = \lambda\nu + \sum_{\omega\in\Omega}\sum_{t\in\mathcal{T}} \pi_\omega y_\omega^t \tag{20}$$

$s.t.$

$$\sum_{\omega\in\Omega_u} y_\omega^t \geq 1 \qquad \forall u \in \mathcal{U}, t \in \mathcal{T}_u \tag{21}$$

$$\sum_{\omega\in\Omega} y_\omega^t \leq \nu \qquad \forall t \in \mathcal{T} \tag{22}$$

$$y_\omega^t \in \{0,1\} \qquad \forall \omega \in \Omega, t \in \mathcal{T} \tag{23}$$

$$\nu \in \mathbb{N} \tag{24}$$

The objective function (20) is the same as Equation (1). Constraints (21) are set covering constraints for demand satisfaction. Constraints (22) enforce the number of vehicles to be less than or equal to ν at each period.

The MP-DARP model is separable with respect to the variables y_ω^t. This set covering formulation is equivalent to $|\mathcal{T}|$ independent DARP route based formulations.

It can be solved to optimality by a solver provided the number of routes $|\Omega|$ remains reasonable. Rather than directly using a solver, we first run the Large Neighborhood Search (LNS) method proposed by Tellez et al. [37] in order to quickly generate good solutions with feasible routes of minimal duration. For the sake of completeness, we summarize the content of this previous contribution: The LNS algorithm iteratively destroys and repairs parts of the current solution using several operators. It is hybridized with a set partitioning component. This component solves a set partitioning problem to reassemble routes that were generated at distinct iterations of the LNS. A reactive mechanism automatically adjusts its parameters. On the LNS side, the classical "best insertion" and "k-regret" operators are used to repair solutions. To destroy a solution, we found that, in combination with the set partitioning component, only those two destroy operators were needed. These are the *random removal* and *history node-pair removal operators*. The paper investigates the use of reconfigurable vehicles, which facilitates the transportation of passengers with wheelchairs. An efficient feasibility checking algorithm is proposed to handle this

feature in repair operators. The route scheduling algorithm minimizes the route duration and checks time windows and maximal ride time constraints. The method was evaluated on real-life instances as well as academic instances of the DARP and was shown to be competitive with other state-of-the-art metaheuristics.

We use the LNS algorithm as a black box that is able to generate a number of good solutions to the DARP. These solutions contains routes that we store in a set $\Omega' \subset \Omega$. The MP-DARP is then solved with Ω' instead of the untractable set Ω. Each time Ω' is updated, dominated routes (i.e. routes $\omega \in \Omega'$ which users are included in another route $\omega' \in \Omega'$ and which cost is greater than ω') are filtered out.

Given that the MP-DARP is a set covering formulation, a solution S may contain users' demands served by more than one route. In this case, removing one of the duplicated visits reduces the solution cost (the triangular inequality is assumed).

A solution S is repaired by updating Ω' as follows. For each user u visited more than once in S and for each route $\omega \in \Omega'$ that visits user u, a new route ω' identical to route ω but that does not visit user u is added to Ω'. Then, the MP-DARP formulation is solved again on the updated set of routes Ω' and a warm-start on S.

4.4 Selection of routes: function $selectRoutes(\mathcal{L}, N, l_{memory})$

When the current pool of routes $\mathcal{L}$ or $\mathcal{L}_{new}$ is too large, the MILP solver cannot improve the given initial solution. Hence, it becomes necessary to select a subset of routes of reasonable size. The function *selectRoutes*, called at lines 9 and 13 of Algorithm 2 is used to randomly select N from a larger set $\mathcal{L}$. This process uses the set l_{memory} of the routes previously selected at former calls of this function.

First, all routes in $\mathcal{L}$ are given a score. When the objective function is f, this score of a route corresponds to the cost of the best TC-DARP solution found so far that uses this route in at least one period. When the objective function is $\boldsymbol{g}$, the score of a route is the consistency measure among all TC-DARP solutions found so far that use this route in at least one period. Then, $\mathcal{L}$ is sorted according to these scores in non-decreasing order.

The route selection process browses the sorted pool of routes. For each route, we check whether it was already selected in a previous call. In this case, it is selected again with a probability γ. Otherwise, it is systematically selected. The process stops when N routes have been selected.

Figure 4 shows an example of the route selection with $N = 6$. Each line represents the sorted pool of routes. Each square represents a route in this sorted pool. The first call selects the first six routes. In the second call, the grey area represents routes that have been selected at the first call. The second call randomly selects 1 route in the grey area and completes the selection with the first five routes in the white area. The third call randomly selects two routes in the grey area and completes the selection with the first four routes in the white area.

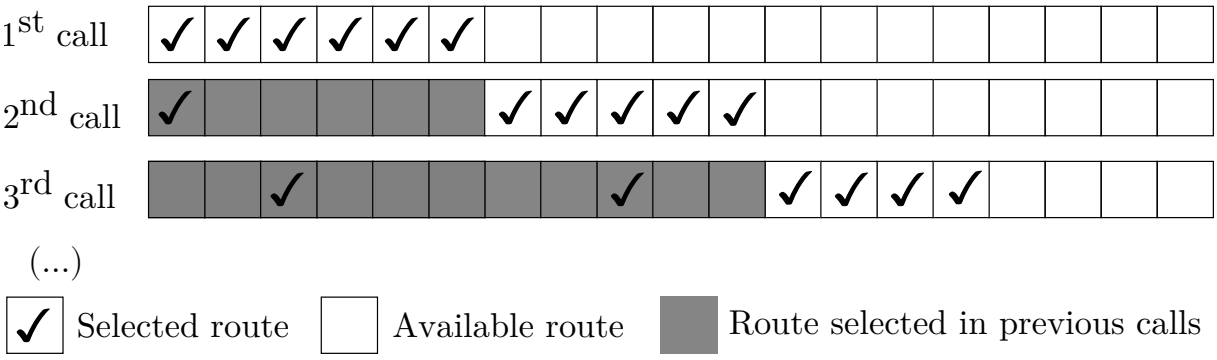


Figure 4: Example of 3 successive calls to the sequential route selection with $N = 6$.

The sequential route selection process increases the probability of selecting together routes that have been part of the same solution with the same overall cost-based or consistency-based performance. The routes must be sorted each time the sequential route selection is called, as performance indicators may change from one iteration to another.

The practical implementation of the *selectRoute* function includes an extension of the N selected routes: the set of selected routes is augmented by *projection* and *complementary routes*, defined by Definition 4.1.

Definition 4.1. Projection and complementarity A route ω' is called a *projection* of a route ω in period $t \in \mathcal{T}$ if it contains only the users of ω who have a demand in period t, in the same sequence as in ω. The route $\omega'' = \omega \setminus \omega'$ is called the *complementary* route of ω'.

The definition 4.1 is illustrated by Figure 5. At period t, Route ω starts from depot D, serves user requests $1, 2$ and 3 and returns to the depot. If users 1 and 2 have a transportation request at period t' and user 3 does not, the route ω' serving user requests 1 and 2 is the projection of ω and the route ω'' serving request 3 is the complementary of ω'.

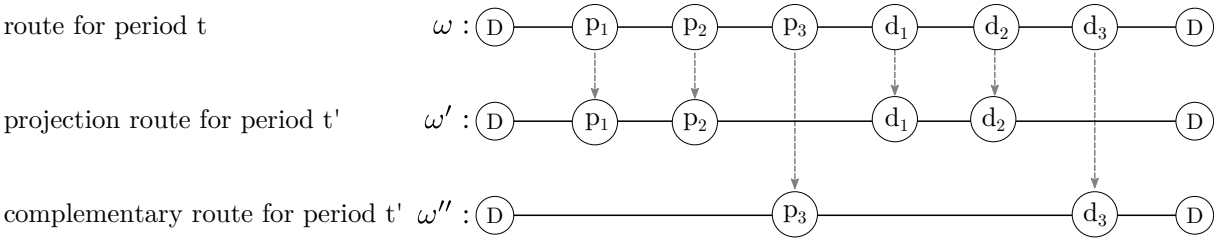


Figure 5: Example of projection and complementary routes

5 Generation of new routes: the DARP with multiple time windows

In section 4.3, it is explained how the SPεC framework is initialized with routes that are generated by solving independent DARP instances with an LNS algorithm. These routes are combined in order to form more consistent solutions. In Algorithm 2 (line 11), the time consistency can nevertheless be improved by generating new routes, denoted $\mathcal{L}_{new}$, that are not likely to be generated by the LNS operators.

This section introduces a new optimization problem: the dial-a-ride problem with multiple time windows (DARPmTW). To the best of our knowledge, this problem has never been treated in the literature. In order to better explain how the the DARPmTW arises from the TC-DARP, we first give an example of multiple time windows for one particular user in Section 5.1. In section 5.2 we explain how times windows are generated for all users and how these time windows are embedded in DARPmTW instances. Section 5.3 focuses on the solution methods to solve the DARPmTW.

5.1 Multiple time windows for a given user

Let us consider a user $u \in \mathcal{U}$ that must be visited in the time window $[a_{p_u}, b_{p_u}]$. In the current solution, the variables h_u^t represent the service time of user u at time period $t \in \mathcal{T}_u$. The set of service times of user u is denoted $\mathcal{H}_u$ and its number of time classes is denoted by $\mu(\mathcal{H}_u)$.

Figure 6 represents the user's schedule. The horizontal axis represents the time window $[a_{p_u}, b_{p_u}]$ and the five service times that can be regrouped in 3 time classes of width Λ: class 1 includes h_u^1, h_u^2 and h_u^3 while isolated values h_t^4 and h_t^5 define two time classes on their own. Note that the service times are regrouped in time classes with the same greedy algorithm as in Feillet et al. [9].

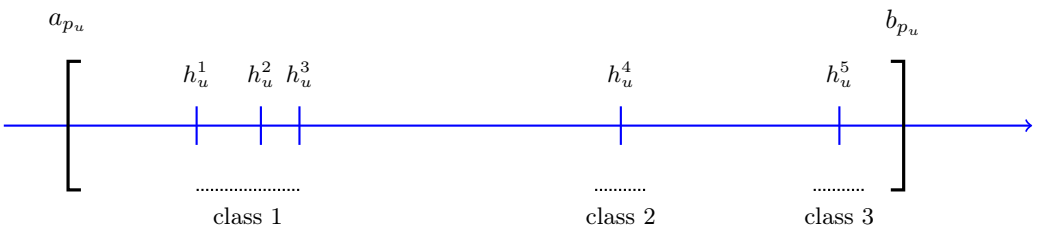


Figure 6: Initial schedule of user u

Assume that, for any reason, the value of h_u^5 has to be modified. The value h_u^5 is first removed from the user's schedule. The remaining partial set of service times is $\mathcal{H}_u \setminus \{h_u^5\} = \{h_u^1, h_u^2, h_u^3, h_u^4\}$. It can be observed that the number of time classes of user u is decreased by 1 if the service time h_u^5 is removed, i.e. $\mu(\mathcal{H}_u \setminus \{h_u^5\}) = \mu(\mathcal{H}_u) - 1$.

Then, depending on the future value of h_u^5, user u will have two or three time classes. This is illustrated by Figure 7. Any value of h_u^5 in the interval $[h_u^3 - \Lambda, h_u^1 + \Lambda]$ is compatible with class 1. Similarly, any value of h_u^5 in the interval $[h_u^4 - \Lambda, h_u^4 + \Lambda]$ is compatible with class 2. Any value out of these intervals creates a class 3.

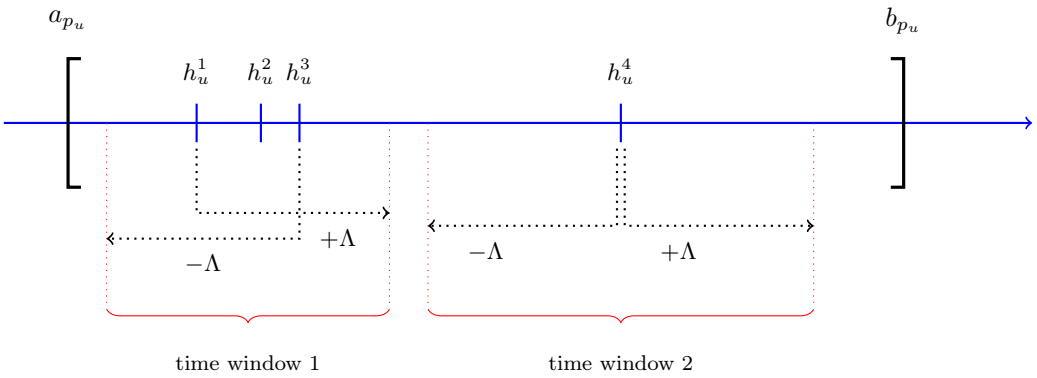


Figure 7: Construction of time windows of user u with $\mathcal{H} = \{h_u^1, h_u^2, h_u^3, h_u^4\}$.

As a consequence, if the objective is to have at most two time classes for user u, the value of h_u^5 must be defined within times windows $[h_u^3 - \Lambda, h_u^1 + \Lambda]$ or $[h_u^4 - \Lambda, h_u^4 + \Lambda]$. If the objective is to have at most three time classes, h_u^5 can take any value in the interval $[a_{p_u}, b_{p_u}]$.

Given a user u and a partial set $\mathcal{H}_u$ of service times, we call $genMTW(\mathcal{H}_u, u)$ the function that returns a set of multiple time windows. The algorithm implementing the construction of multiple time windows is detailed in Appendix A.1.

5.2 Multiple time windows for the DARPmTW

Section 5.1 presented an example of multiple time windows definition for some user u given a known set of service times $\mathcal{H}_u = \{h_u^1, h_u^2, h_u^3, h_u^4\}$. In this section, we define multiple time windows, denoted by $\mathcal{W}_u^t$, for any user $u \in \mathcal{U}$ and for any set $\mathcal{H}_u$. More precisely, we explain how to define the parameters of the function $genMTW(\mathcal{H}_u, u)$.

In order to increase consistency and to generate routes that were not in the previous pool $\mathcal{L}$, we propose a very large neighborhood operator that sequentially removes the value of the variables associated with each time period and rebuilds a feasible solution. At each time period, the operator focuses on a particular user, denoted by $\bar{u}$, which is randomly selected among all users with $C_{\max}$ time classes. The idea is to rebuild a solution that decreases the number $\mu(\mathcal{H}_{\bar{u}})$ of time classes of user $\bar{u}$ and controls the number $\mu(\mathcal{H}_u)$ of time classes for all other users.

Given one period $t \in \mathcal{T}$, the current value of variables h_u^t is "forgotten" (removed) and multiple time windows $\mathcal{W}_u^t$ are set according to the four cases represented in Table 5. These cases are defined according to the answer to the following questions: i) Is the selected user $\bar{u}$? ii) Is the number of time classes of user u decreased by 1 when the service time h_u^t at period t is removed? This question can be answered by checking if the inequality $\mu(\mathcal{H}_u \setminus \{h_u^t\}) < \mu(\mathcal{H}_u)$ holds.

The yes/no answers to these questions yield four ways to define multiple time windows, representing by the four main cells of Table 5.

Note that using $\mathcal{W}_u^t = \{[a_{p_u}, b_{p_u}]\}$ in the bottom right-hand case may result in an increase of time classes for user u. We observed that this temporary relaxation helps decrease the number of time classes of user $\bar{u}$. The idea of relying on a randomly selected user $\bar{u}$ was originally proposed by [9] to solve a time consistent VRP. The contribution is to extend the approach of [9] from a VRP to a DARP. Finding the best routes that satisfy the multiple time windows amounts to solve a Dial-A-Ride Problem with multiple Time Windows (DARPmTW).

Algorithm 3 describes the process used to generate new routes. It focuses on a randomly selected user $\bar{u}$ with $C_{\max}$ time classes (lines 2). For each time period $t \in \mathcal{T}$, multiple time windows for user $\bar{u}$ are sought such that its number of time classes will be decreased (line 4). For all other users, multiple time windows are sought such that their number of time classes will be controlled (line 6), as detailed in Table 5. These time windows are used to define a new instance of the DARPmTW for each time period. Solving these instances yields a set of routes denoted by $\mathcal{L}_{new}^t$ for each time period t (line 8). The sets

Is u the selected user $\bar{u}$?	Is the number of time classes of user u decreased by 1 when the service time h_u^t at period t is removed? **Yes**	**No**
Yes	Changing the service time $h_{\bar{u}}^t$ to any service time in the time windows $\mathcal{W}_{\bar{u}}^t$ has to decrease the number of time classes for this user: $\mathcal{W}_{\bar{u}}^t = genMTW(\mathcal{H}_{\bar{u}} \setminus \{h_{\bar{u}}^t\}, \bar{u})$.	Let $[e_{\bar{u}}^t, l_{\bar{u}}^t] \in \mathcal{W}(\mathcal{H}_{\bar{u}}, \bar{u})$ be the time window satisfied by service time $h_{\bar{u}}^t$ ($e_{\bar{u}}^t \leq h_{\bar{u}}^t \leq l_{\bar{u}}^t$). Defining $\mathcal{W}_{\bar{u}}^t = genMTW(\mathcal{H}_{\bar{u}}, \bar{u}) \setminus \{[e_{\bar{u}}^t, l_{\bar{u}}^t]\}$ will enforce a decrease in the number of time classes for this user.
No	The service time h_u^t defines a time class on its own and any feasible service time at period t can be accepted: $\mathcal{W}_u^t = \{[a_{p_u}, b_{p_u}]\}$.	$\mathcal{W}_{\bar{u}}^t = genMTW(\mathcal{H}_{\bar{u}}, \bar{u}) \setminus \{[e_{\bar{u}}^t, l_{\bar{u}}^t]\}$, with a probability $\nu = \theta \times (C_{\max} - \mu(\mathcal{H}_u))$, where θ is a fixed parameter. Otherwise, $\mathcal{W}_u^t = \{[a_{p_u}, b_{p_u}]\}$.

Table 5: Overview of cases for the definition of multiple time windows $\mathcal{W}_u^t$.

$\mathcal{L}_{new}^t$ are finally merged (line 10) and returned. In section 5.3 we detail the solution method used to solve the DARPmTW instances (line 8).

Algorithm 3: Fonction generateNewRoutes(S)

Arguments: S: a feasible solution of the consistent DARP
Result: list of routes L_{new}

1 Initialize C_{max} to the maximum number of time classes in S
2 Randomly select a user $\bar{u}$ with $C_{\max}$ time classes
3 **for** *all time periods* $t \in \mathcal{T}$ **do**
4 Define multiple time windows $\mathcal{W}_{\bar{u}}^t$ for user $\bar{u}$ as detailed in Table 5
5 **for** *all users* $u \neq \bar{u}$ **do**
6 Define multiple time windows $\mathcal{W}_u^t$ as detailed in Table 5
7 **end**
8 $\mathcal{L}_{new}^t \leftarrow$ solveDARPmTW($\mathcal{W}_u^t$) `/* See Section 5.3 */`
9 **end**
10 $\mathcal{L}_{new} \leftarrow \cup_{t \in \mathcal{T}} \mathcal{L}_{new}^t$
11 **return** $\mathcal{L}_{new}$

5.3 Solving the DARPmTW

Line 8 of Algorithm 3 consists of solving a DARP with multiple Time Windows (DARPmTW). Additionally, the duration of routes should be minimized: waiting times caused by time windows are authorized but waiting times aimed at artificially improving time consistency are forbidden.

To solve this optimization problem, we reuse the LNS-SCP used in Section 4.3 to solve single period DARPs with one time window at each pickup or delivery point. However, this LNS-SCP requires two adaptations. First, since the SCP component is somewhat redundant with the set partitioning problems solved in SPεC, we do not activate it here. The algorithm is therefore called LNS in the following. Second, a key point of the LNS is to handle the route duration minimization when repairing destroyed solutions. This amounts to check a set of temporal constraints through a route scheduling algorithm, called *the DARPmTW scheduling algorithm*.

More specifically, let us consider the feasibility check of an insertion in a route $\omega \in \Omega$. For each user

u visited by the route ω, the DARPmTW scheduling algorithm checks temporal constraints: a maximum ride time $\bar{T}_u$, a pickup time window $[a_{p_u}, b_{p_u}]$ and a delivery time window $[a_{d_u}, b_{d_u}]$. If these temporal constraints are satisfied, a service time h_i is computed for each node such that: (i) the route duration is minimized and (ii) service times are scheduled as early as possible at each pickup node, delivery node or depot. The DARPmTW scheduling algorithm computes a *maximum route time shift* Δ_ω^+. This value indicates how much the route schedule can be shifted forward, while preserving both its feasibility and the route duration. Hence, shifting the starting time of a route by a value $0 \leq \delta \leq \Delta_\omega^+$ shifts the service times of all vertices on the route by the same value without impacting the route duration nor waiting times. The detailed procedure, proposed in Tellez et al. [37], can be found in Appendix A.2.

Once the pickup time window $[a_{p_u}, b_{p_u}]$, the delivery time window $[a_{d_u}, b_{d_u}]$ and ride time constraints $\bar{T}_u$ are satisfied, we check the satisfaction of the multiple time windows $\mathcal{W}_u^t$ with a new procedure. Specifically, this procedure checks if there is a service time for each user u compatible with a time window in $\mathcal{W}_u^t$. Given a route $\omega \in \Omega$ with a minimal duration and service times scheduled as early as possible, Algorithm 4 checks if each user u on this route can be picked up within a set of multiple time windows $\mathcal{W}_u^t$ without increasing the duration of the route. Accordingly, the only variable in this algorithm is the departure time of the route, which can be postponed by a value $0 \leq \delta \leq \Delta_\omega^+$ called *route time shift*. Note that, for morning routes, the service in multiple time windows is only verified at pickups because time consistency is measured at these nodes. For afternoon routes, time consistency must be checked at delivery points only.

Algorithm 4: DARPmTW scheduling Algorithm

Parameters: a route ω; $\mathcal{U}_\omega$: set of users ordered by non-decreasing pickup times; $\mathcal{W}_u^t$: set of multiple time windows sorted in non-decreasing order of the earliest value; h_u: earliest service time for the pickup of user u given the duration of route ω is minimal; Δ_ω^+: maximum route time shift of route ω.

Output: if route ω is **feasible** or **unfeasible**

```
1  δ ← 0                                                    /* the route time shift */
2  for u ∈ U_ω do
3      scheduledUser ← false
4      while ¬scheduledUser and W_u^t ≠ ∅ do
5          [a, b] ← first time window in W_u^t
6          if h_u + δ > b then                              /* the time window is too early */
7              remove first time window [a, b] from W_u^t
8          else if a ≤ h_u + δ ≤ b then                     /* the time window is satisfied */
9              scheduledUser ← true
10         else if h_u + δ < a then    /* the departure of ω should be delayed to meet the
               time window */
11             δ ← a − h_u
12             if δ ≤ Δ_ω^+ then
13                 jump to line 2: the for loop is restarted to the first user u in U_ω
14     if ¬scheduledUser then
15         return unfeasible
16 return feasible
```

Let $\mathcal{U}_\omega$ be the set of users ranked in non-decreasing order of their pickup service time. Algorithm 4 looks for a route time shift $\delta \in [0, \Delta_\omega^+]$ such that for all users $u \in \mathcal{U}_\omega$ there exists a time window $[a, b] \in \mathcal{W}_u^t$ in which the shifted service time $h_u + \delta$ can be scheduled. Note that all nodes in route are shifted forward by the same quantity of time. Thus, it is not possible to increase the duration of the route in order to ensure feasibility.

In line 1, the route time shift δ is initialized to 0. Users are considered sequentially (line 2). For each user $u \in \mathcal{U}$, its first time window $[a, b]$ is evaluated (lines 4-5). Three cases are considered:

(i) If the shifted service time takes place after the end of the time window ($h_u + \delta > b$), the time window $[a, b]$ can never be satisfied: it is removed from set $\mathcal{W}_u^t$ (lines 6-7). The next iteration of the *while* loop will directly check the next time window for user u.

(ii) If the shifted service time takes place in time window $[a, b]$, the shifted service time is feasible for

the pickup of user u (lines 8-9). The algorithm continues with the next user.

(iii) If the shifted service time takes place before the opening of the time window ($h_u + \delta < a$, line 10), the route shift δ has to be increased to $a - h_u$, so that the new shifted service time is exactly a (line 11). At this point, two cases are possible a) the new value of route time shift δ is feasible (i.e. $\delta \leq \Delta_\omega^+$), and the main loop is restarted from the first user with the new value of δ (jump from line 13 to line 2); b) δ is larger than the maximum allowed shift Δ_ω^+ and the route is infeasible (line 15).

Finally, if each user has a feasible time window given the same time shift δ, then the route is declared feasible (line 16).

In the worst case, all time windows are removed ($\sum_{u \in \mathcal{U}_\omega} |\mathcal{W}_u^t|$ operations). For each removal, the route time shift is increased and then the procedure is restarted. One iteration of the main loop (2–13) cannot take more than $|\mathcal{U}_\omega|$ operations. So the worse-case time complexity of Algorithm 4 is $\mathcal{O}(|\mathcal{U}_\omega| \times \sum_{u \in \mathcal{U}_\omega} |\mathcal{W}_u^t|)$.

6 Computational experiments

The matheuristic described in Section 4.1 was coded in C++ and the mathematical models were solved with CPLEX Concert Technology 12.6 running on a single thread on an Intel Xeon E5-1620 v3 @3.5Ghz processor.

This section details computational experiments in two families of instances. It is structured as follows: Section 6.1 presents the value of parameters used by our algorithms. Section 6.2 introduces the instances used to evaluate our approach. They are built from real data provided by the Synergihp Rhône-Alpes Company. In Section 6.3, SPεC is assessed on benchmark instances of [11] and [9] for the time consistent VRP. Finally, Section 6.4 presents managerial insights regarding cost performance and time consistency.

6.1 Parameter settings

After preliminary tests on a representative subset of instances, parameters shown in Table 6 were found to provide the best average performance.

Global parameters	
$\varepsilon = 0.01$	initial value of epsilon
$\phi = 1.5$	epsilon increase factor
$MaxIter = 4$	maximum number of iterations without improvement
$N = 100$	number of routes appended to the pool at each call of the TC-DARP (instances with less than 250 users).
$N = 200$	number of routes appended to the pool at each call of the TC-DARP (instances with more than 250 users)
$t_{\max} = 60 \times \lceil \lvert\mathcal{U}\rvert/100 \rceil$s	MILP solver time limit for instances with less than 250 users
$t_{\max} = 540s$	MILP solver time limit for instances with 250 users
$N_{\max} = 50000$	size of the pool $\mathcal{L}$
$N_{new} = 5000$	maximum number of routes in $\mathcal{L}_{new}$
Route selection parameters	
$\theta = 10\%$	relaxation parameter in DARPmTW
$\gamma = 10\%$	percentage of routes that can be re-selected in the sequential route selection
$\rho = 6$	Random Biased Selection parameter

Table 6: Summary of parameters of the SPεC algorithm

The value of parameter ε has a strong impact on the computing time. Higher values of ε help to reduce the computation time. However, the quality of the Pareto front approximation is considerably deteriorated. Thus, $\varepsilon = 0.01$ was taken as a good trade-off between computing time and quality of the solution. SPεC is less sensitive to parameter ϕ but its value needs to be greater than 1.5 to have significant impact on the value of ε.

We found that a value of $MaxIter$ greater than 4 does not improve the quality of each point of the Pareto front approximation. Parameters N and $t_{\max}$ were determined in order to maximize the number of times when the MILP solver is able to solve the TC-DARPs to proven optimality.

The TC-DARP considers two pools of routes: $\mathcal{L}$ and $\mathcal{L}_{new}$. In order to keep the number of routes in memory under control, limits in the maximum size of the pool $\mathcal{L}$ and of the $\mathcal{L}_{new}$ were set to $N_{max} = 50000$ and $N_{new} = 5000$, respectively.

We compared several settings of the algorithm in order to assess its key components. In particular, we run a variant of the SPεC algorithm without the procedure presented in Section 4.4 to generate new routes. This resulted in a slight cost increase on solutions with 2 time classes and more, and a 10% cost increase on solutions with 1 time class. Moreover, the hypervolume indicator increases by 17% if the generation of new routes is disabled.

We also implemented a randomized route selection based on the roulette wheel mechanism inspired by [26]. This randomization did not bring significant improvement. A detailed evaluation of the algorithm's components can be found in [36].

6.2 Description of instances

The time consistent DARP studied in this paper arises in the context of transportation of people with disabilities. We collected real data from the Synergihp Rhône-Alpes Company based in Lyon, France. This data concerns the transportation of hundreds of users to MSIs.

From Monday to Friday, the users are carried in the morning from their home to a MSI. In the afternoon, they are driven back home. Without loss of generality, this paper presents the results of our research for morning trips.

The data collected from Synergihp includes 575 users from a large geographical area around the city of Lyon. We partitioned this data in two independent geographical areas (distinct users, distinct MSIs), yielding two large instances with with 280 to 295 users, respectively. These two large instances were in turn partitioned into 4 medium-size instances with 120 to 160 users. Finally, the four medium-size instances were partitioned into 8 small-size instances with 60 to 80 users. We assume an infinite homogeneous fleet of vehicles with a capacity of 4 seats and 3 wheelchair spaces. Vehicle costs provided by the company include an hourly cost α =23.8€ and a cost per kilometer τ =0.17€. No fixed vehicle cost λ was given by the company. Moreover, with a homogeneous fleet of vehicles, this cost has no major impact on the solution nor on solutions consistency. Thus, we simply assume an arbitrarily small value $\lambda = 1$ in order to favor solutions with similar route cost but fewer vehicles.

Travel times and distances are obtained from the Open Source Routing Machine[2] (OSRM) proposed by Luxen and Vetter [20]. For each user $u \in \mathcal{U}$, we define maximum ride times according to direct travel time t_{p_u,d_u} between the pickup location p_u and the delivery location d_u. The following formula generates maximum ride times (RT) that are between 15 and 30 minutes longer than direct travel times: $RT = 15 \times \lceil (t_{p_u,d_u} + 15)/15 \rceil$.

Time windows at MSIs are 15 minutes wide. The size of time classes is 10 minutes wide. Finally, time windows at pickup locations and service times strongly influence the actual design of routes, but they have no impact on the efficiency of our solution method. We therefore ignored them for the sake of simplicity.

6.3 Performance evaluation on benchmarks from the literature

As the TC-DARP is a new problem, there is no benchmark in the literature. However, to evaluate the performance of SPεC, we solve reference instances for two other time consistent routing problems. The first benchmark is an adaptation of the conVRP instances of Groër et al. [11]. This adaptation, proposed by Feillet et al. [9], transforms the small conVRP instances of Groër et al. [11] into TC-VRP instances. These instances are denoted RconVRP. They include up to 12 users over 3 days and have been solved to optimality by a MILP solver. The second benchmark set is the TC-VRP from Feillet et al. [9]. It contains instances for up to 65 users over 5 time periods.

Although the TC-VRP is the closest problem to the TC-DARP, there are some differences between both problems. The TC-VRP has the following hypothesis: (i) it has a single depot and no time windows; (ii) it assumes a limited fleet of vehicles; (iii) the consistency objective function is the maximal number of time classes over all users (i.e. $C_{\max}$); and (iv) routes must start at time 0, with no waiting time allowed. To be solved by SPεC, TC-VRP instances have been converted to TC-DARP instances by defining one copy of the depot for each request. Ride times and time windows have been relaxed by setting arbitrary large values. Finally, since the VRP routes are not subject to time windows nor ride times, routes can

[2] `http://project-osrm.org/`

be traveled in either direction. Thus, each time a route is appended to the pool, the reverse route is also appended.

6.3.1 RconVRP instances

This benchmark proposes 10 small instances of the TC-VRP: the first 5 instances with 10 users and the next 5 with 12 users. Instances have been solved to optimality with CPLEX. Table 7 presents the comparison with SPεC over 10 runs.

Instance	Opt. Cost			SPεC		
$C_{\max}$	$\leq$3	$\leq$2	1	$\leq$3	$\leq$2	1
RconVRP10-1	92.91	92.91	92.91	**92.91**	**92.91**	**92.91**
RconVRP10-2	80.42	80.42	80.96	**80.42**	**80.42**	82.83
RconVRP10-3	94.12	94.12	94.37	**94.12**	**94.12**	**94.37**
RconVRP10-4	93.71	93.71	94.09	**93.71**	**93.71**	**94.09**
RconVRP10-5	83.84	83.84	96.01	**83.84**	84.50	96.70
RconVRP12-1	103.65	103.65	104.40	**103.65**	**103.65**	109.19
RconVRP12-2	73.89	73.89	81.25	**73.89**	**73.89**	83.40
RconVRP12-3	82.77	82.77	83.12	**82.77**	**82.77**	**83.12**
RconVRP12-4	97.57	97.57	101.91	**97.57**	98.55	104.31
RconVRP12-5	83.63	83.63	89.25	**83.63**	**83.63**	91.38
			Avg Gap (%)	0.0%	0.2%	1.5%

Table 7: Benchmark on RconVRP instances reported in [9]

Columns 2-4 (Opt Cost), present the cost of optimal solutions for each time class. In the next 3 columns, we report the average performance of SPεC for each time class. Note that SPεC could not find most of the solutions with 3 time classes and one with 2 time classes. For each instance, the Gap is computed as (Avg Cost - Best) / Best $\times$ 100. The last row is the average gap (Avg Gap) overall instances which is the average value of the corresponding column.

For single time class solutions, SPεC founds optimal solutions in the 10 runs for 4 instances of 10. The average gap is 1.5%. The performance is better with 2 time classes for which 7 of 10 solutions are optimal and with an average gap of 0.2%.

6.3.2 Results on time consistent VRP instances

This benchmark was built from real data collected in 14 distinct MSIs, with a number of users ranging from 15 to 65, and a number of time periods equal to 5 (Monday to Friday). For each MSI, 5 profiles of transportation demands where randomly generated where each profile corresponds to the percentage of people present during the complete week. This percentage varies between 50% and 90%. This yields a total of 70 benchmark instances.

Transportation cost of solutions with 1 to 5 time classes are provided for most instances. Feillet et al. [9] solved the TC-VRP with an LNS-based matheuristic with a time limit of 1 hour. SPεC stops when all users reach a single time class. For each value of C_{max}, our lexicographic optimization explores all non-dominated solutions. This approach is more time consuming but returns a more complete Pareto front approximation that can help decision makers to select intermediate trade-off solutions for each time class.

Tables 8 and 9 show the average gap of SPεC with respect to the LNS method of Feillet et al. [9], aggregated in two different ways. For each instance, we compute the gap as (Cost SPεC$-$Cost LNS)/Cost LNS $\times$ 100. Thus, any negative gap represents an improvement.

Table 8 shows the numerical results aggregated by percentage of presence during the week. For example, data-5-Y aggregates instances where, on average, 50% of users are transported everyday, while in the group data-9-Y the average percentage of users transported rises to 90%.

The average relative gap (Avg Gap) overall instances between the results obtained with SPεC and the LNS was improved for all time classes. However, slightly better results are reported for solutions with 3, 4 and 5 time classes. This result is confirmed with the number of new best-known solutions (Nb new BKS) which is more than 50 for solutions above 3 time classes, and 35 for solutions with 1 and 2 time

Instance	Avg Gap Transportation cost				
$C_{\max}$	≤ 5	≤ 4	≤ 3	≤ 2	1
data5-Y	**-1.10%**	**-1.00%**	**-1.02%**	**-0.21%**	**-0.37%**
data6-Y	**-1.03%**	**-1.03%**	**-0.81%**	**-0.26%**	0.83%
data7-Y	**-1.09%**	**-0.96%**	**-0.86%**	0.13%	**-0.13%**
data8-Y	**-1.06%**	**-1.00%**	**-0.72%**	0.00%	**-0.25%**
data9-Y	**-0.61%**	**-0.61%**	**-0.49%**	**-0.27%**	**-1.76%**
Avg Gap (%)	**-0.98%**	**-0.92%**	**-0.78%**	**-0.12%**	**-0.33%**
Nb Sols	70	70	70	70	70
Nb new BKS	63	59	53	35	35

Table 8: Results aggregated by percentage of user requests on the benchmark of Feillet et al. [9]

classes. A total number of 245 strictly new best solutions were found, as shown on the last row of the table. Detailed results can be found in Appendix B.

Table 9 shows the numerical results aggregated by MSI. The last two digits of the instance name represent the number of users. This table shows that SPεC has better performance with larger instances. However, dataX-59 instances are in particular the most difficult to solve for SPεC, with an extra cost of 3.41% for $C_{\max} = 2$ solutions and 2.39% for $C_{\max} = 1$ solutions.

Instance	Avg Gap Transportation cost				
$C_{\max}$	≤ 5	≤ 4	≤ 3	≤ 2	1
dataX–15	0.03%	0.03%	**0.00%**	0.47%	3.04%
dataX–21	**-0.15%**	0.07%	0.18%	0.24%	1.53%
dataX–25	**-0.21%**	**-0.06%**	0.01%	0.19%	0.53%
dataX–26	**-0.21%**	**-0.21%**	**-0.03%**	0.04%	0.08%
dataX–27	**-0.27%**	**-0.12%**	**-0.16%**	0.03%	**-0.93%**
dataX–32	**-0.54%**	**-0.54%**	**-0.19%**	0.10%	0.49%
dataX–41	**-1.15%**	**-1.15%**	**-1.65%**	**-1.19%**	**-2.46%**
dataX–44	**-0.75%**	**-0.75%**	**-0.61%**	**-0.72%**	**-1.19%**
dataX–46	**-1.05%**	**-1.05%**	**-0.62%**	**-0.18%**	**-0.34%**
dataX–48	**-1.24%**	**-0.98%**	**-0.77%**	**-0.20%**	**-1.38%**
dataX–55	**-1.91%**	**-1.91%**	**-1.63%**	**-0.92%**	**-4.84%**
dataX–59	**-2.17%**	**-2.12%**	**-1.62%**	3.41%	2.39%
dataX–64	**-2.30%**	**-2.30%**	**-2.14%**	**-1.66%**	**-0.18%**
dataX–65	**-1.77%**	**-1.79%**	**-1.71%**	**-1.33%**	**-1.42%**
Avg Gap (%)	**-0.98%**	**-0.92%**	**-0.78%**	**-0.12%**	**-0.33%**

Table 9: Results aggregated by instance size on [9] benchmark

6.4 Managerial insights on time consistency and transportation costs

This section reports managerial insights regarding the relationship between time consistency and transportation costs. Figures 8-10 show the Pareto front approximation obtained on Synergihp Rhône-Alpes instances with 60, 160 and 280 users, respectively. The transportation cost is presented as the percentage of cost increase with respect to the cheapest solution found in that instance (x-axis). The time consistency of non-dominated solutions is shown in a vector form on the vertical axis. Each element of the vector represents the number of users with 3, 2 and 1 time classes on that solution, respectively. Note that solutions with 4 or 5 time classes are not represented because, in our tests, they have always been dominated by a solution with 3 time classes.

These Pareto front approximations provide decision makers with a fine intuition of the cost of time consistency associated with each user. The first finding is that all Pareto front approximations start with the majority of users having a single time class and very few users having 3 time classes. With a minor increase of cost, all users have at most 2 time classes (until the dotted line). This means that an useful

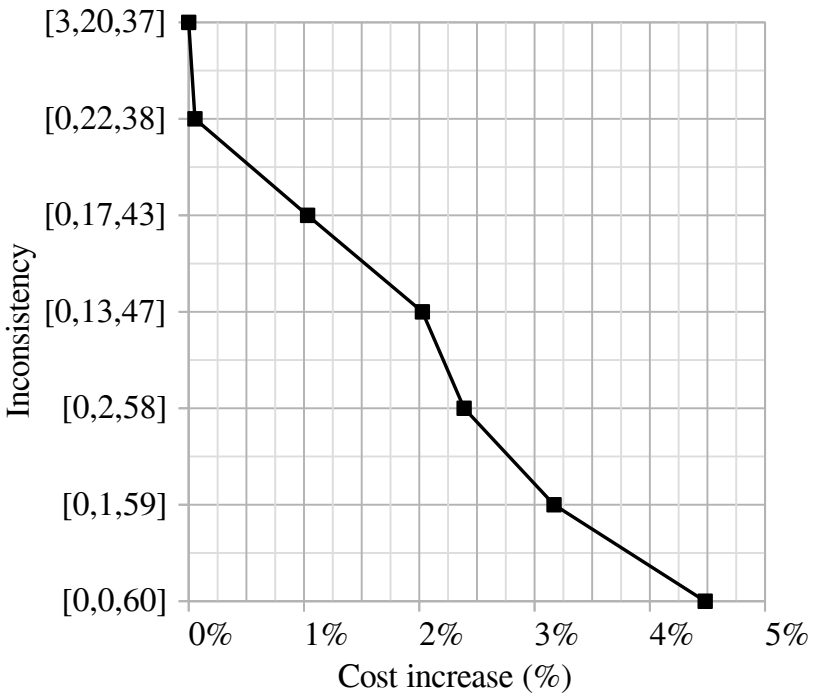


Figure 8: Pareto front approximation for instance TCDARP_01_60

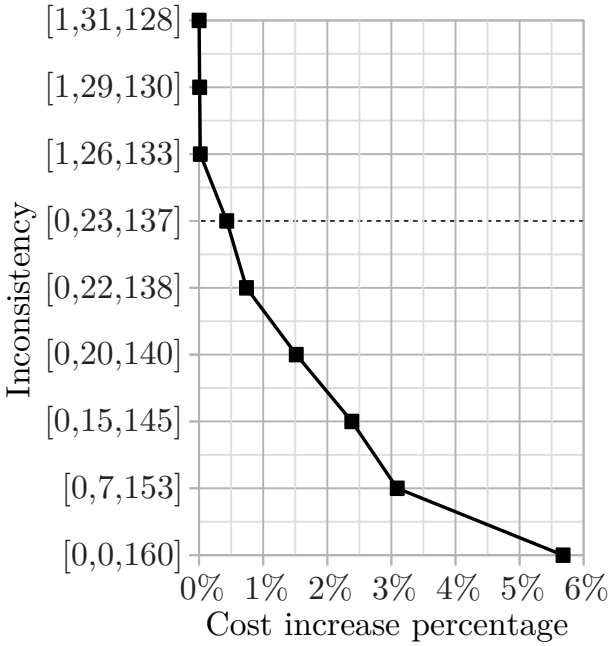


Figure 9: Pareto front approximation for instance TCDARP_10_160

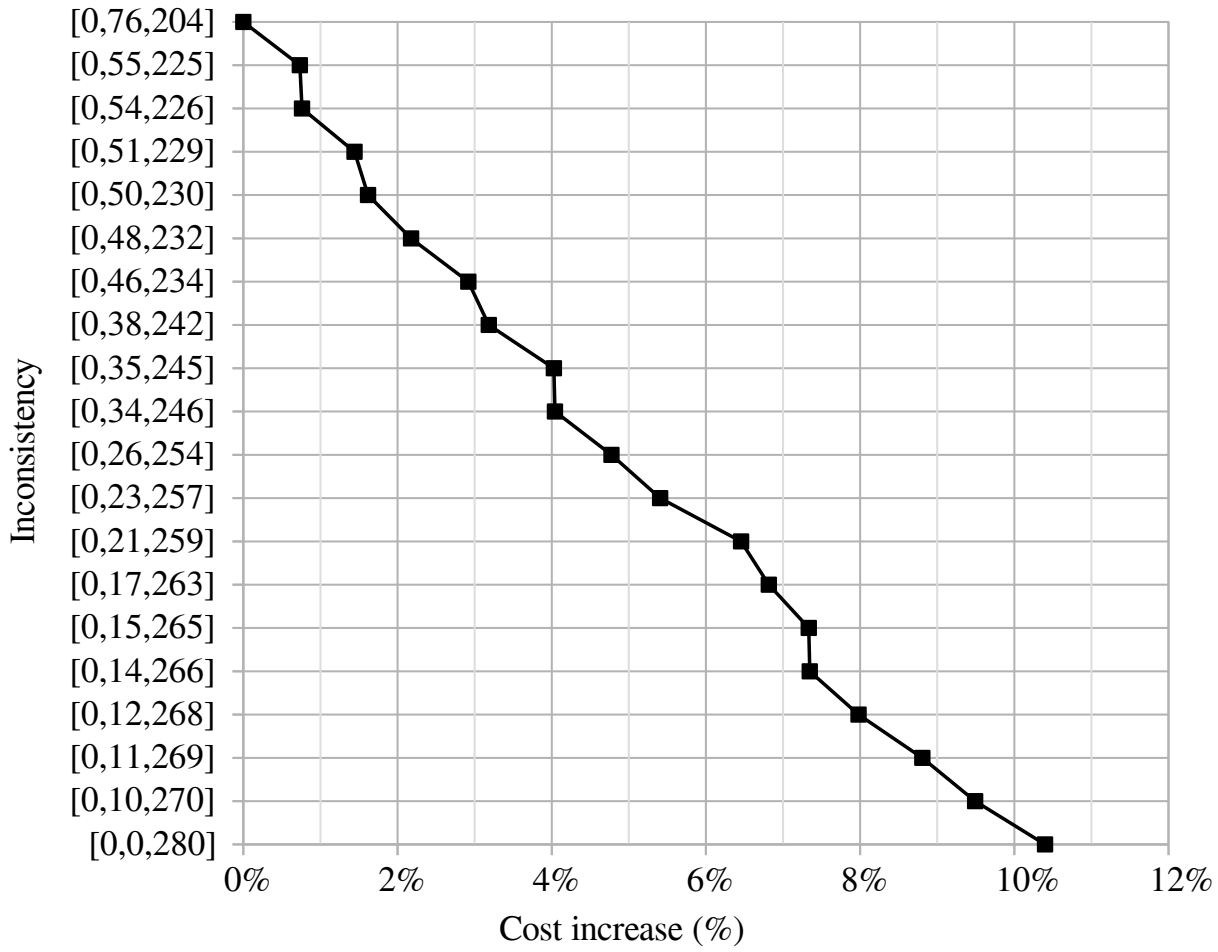


Figure 10: Pareto front approximation for instance $TCDARP\_12\_280$

consistent solution can be found with respect to the cost of the cheapest solution, and a small increase of cost can significantly improve the solution for users with many time classes.

Depending on the instance, the increase of cost for reaching single time class solutions can vary from 1% to 10%. Note that values on the y-axis are ordered but non-scaled as the distance between points is always constant. Figure 11 shows the same instances on a common scale for solutions with a maximum of 2 time classes per user. The y-axis presents the percentage of users with 2 time classes. It shows that each instance has very different trade-offs depending on the size and the consistency level of the solution.

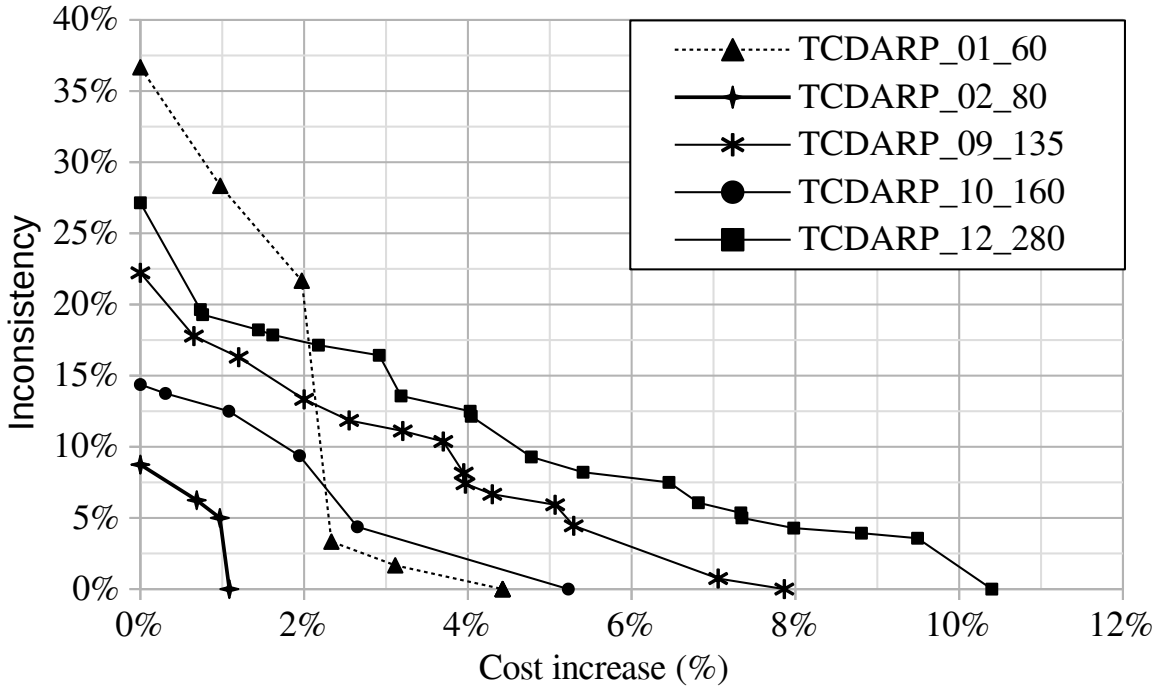


Figure 11: Scaled representation of the Pareto front approximations on five representative instances

Table 10 gives some additional details on the non-dominated solutions shown in Figure 8 (instance TCDARP_01_60). The first column reports the inconsistency vector of each non-dominated solution. Columns 2 to 4 show the cost, the average route duration and the average ride time per user in the corresponding solutions, respectively. The vectors in column 5 represent the number of routes for each time period (from Monday to Friday). The last column reports the total number of routes. This example shows that improving consistency requires increasing the number of routes, which results in a decrease of the average route duration. It is noticeable that the number of routes for a given day does not increase monotonously. For example, the number of routes on Wednesdays is 3 in the first two solutions, then 5 and finally 4. Considering the user convenience perspective. Reducing the number of time classes generally comes with a reduction of the users ride time. Nevertheless, we note that it was necessary to increase the average ride time to find a solution with one time class.

Inconsistency	Cost	Avg route duration (hour)	Avg ride time (min)	Nb. routes per day	Total Nb. of routes
$[3, 20, 37]$	1291.03	2.08	22.97	[4, 4, 3, 5, 3]	19
$[0, 22, 38]$	1291.73	1.97	22.49	[4, 4, 3, 5, 4]	20
$[0, 17, 43]$	1304.34	1.81	21.83	[4, 4, 5, 5, 4]	22
$[0, 13, 47]$	1317.20	1.82	21.30	[4, 4, 5, 5, 4]	22
$[0, 2, 58]$	1321.87	1.82	21.48	[4, 5, 4, 5, 4]	22
$[0, 1, 59]$	1331.94	1.75	21.36	[5, 5, 4, 5, 4]	23
$[0, 0, 60]$	1348.87	1.78	22.00	[5, 5, 4, 5, 4]	23

Table 10: Solution's statistics instance TCDARP_01_60

Figure 12 shows how the consistency requirement concretely modifies the service times for some

users. This figure represents the times at which two users are picked-up in several solutions of instance TCDARP_01_60. The x-axis represents the time line. On each horizontal line, the five weekly service times of users 1 and 2 are represented. Each line represents different non-dominated solutions denoted $s_1, ..., s_5$, in which the users have one, two or three classes.

In solution s_1, user 1 has 3 classes with service times varying from 8:00 to 8:30. One time class (class 3 - around 8:27) gathers 3 services out of 5. In solution s_2, with 2 time classes, 4 service times out of 5 belong to time class 2, centered around 8:28. In solution s_3, all service times are centered around 8:28. A similar phenomenon is observed for user 2. To move from solution s_4 with two time classes to solution s_5 with one time class, the algorithm finds routes to serve the user within the time class with the greatest number of service times in s_4.

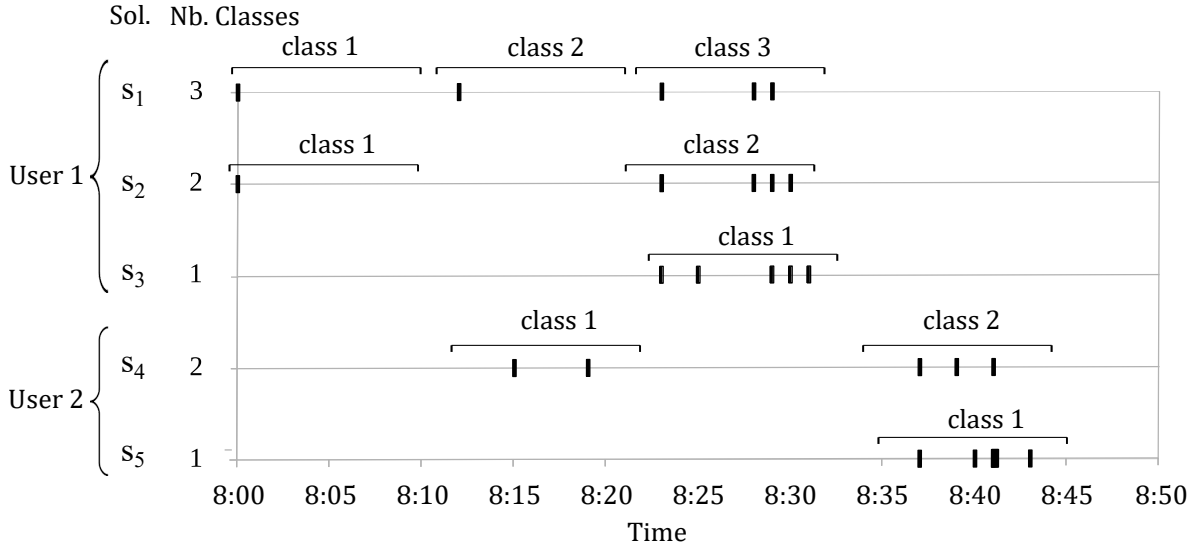


Figure 12: Schedule improvement for 2 users of instance TCDARP_01_60

6.5 Economic impact of shifting route departure times

In this section we compute the impact of having flexible route departure times on cost and time consistency. This effect has been studied by Kovacs et al. [14] for the conVRP, showing that departure time flexibility provides considerable improvement in the solution quality under tight consistency requirements. As far as the TC-DARP is concerned, the departure flexibility is limited by time windows and maximum ride-time constraints. The departure of a route can be scheduled at any time between its earliest and its latest departure date. We define the *maximum time shift* of a route as the difference between these two schedules. The maximum time shift of a route ω is denoted by Δ_ω^+.

Table 11 measures the impact of the departure time shift on the complete set of Synergihp Rhône-Alpes instances. Columns 2–4 present the best results on 5 runs of the SPεC when the route time shift is allowed, for values of C_{max} decreasing from 3 to 1. Missing values in column 2 mean that all solutions found with $C_{max} = 3$ were dominated by another solution with $C_{max} = 2$. Column 5 (%R-Shift) shows the percentage of routes in which a departure time shift is actually implemented. Columns 6–8 present the minimum gap on 5 runs with respect to the minimum cost found by the SPεC when no time shift is allowed ($\Delta_\omega^+ = 0$). The last row shows the average values.

Since the solution method is heuristic, some some small negative gaps can be observed for $\Delta_\omega^+ = 0$. If the solutions found were optimal, negative gaps would not exist. These results show that shifting time departure of routes yields 2.79% savings on average, with values ranging from 0% to 9%. According to these results, time consistency can be achieved at a lower cost when route departure times are not fixed in advance. This also implies that shifting time departure can be a lever to improve time consistency without significantly increasing transportation costs in a DARP context.

Instance	SPεC				SPεC($\Delta_\omega^+ = 0$)		
$C_{\max}$	3	2	1	%R-Shift	3	2	1
TCDARP_00_80	2045.54	2044.43	2061.89	49%	-0.07%	0.25%	0.35%
TCDARP_01_60	1291.03	1291.73	1348.87	40%	0.18%	0.48%	0.20%
TCDARP_02_80	2523.21	2533.86	2561.40	31%	0.00%	-0.01%	0.31%
TCDARP_03_70		1735.07	1752.43	33%		0.00%	0.30%
TCDARP_04_80		1207.72	1220.91	32%		0.00%	2.44%
TCDARP_05_80	1871.25	1871.53	1923.21	40%	0.37%	0.06%	0.08%
TCDARP_06_60		3304.41	3332.21	42%		0.00%	0.54%
TCDARP_07_65	1865.46	1868.15	1920.95	50%	0.00%	-0.07%	3.22%
TCDARP_08_120		5500.45	5534.44	39%		0.02%	1.76%
TCDARP_09_135	2857.18	2880.57	3107.12	38%	0.27%	-0.18%	1.17%
TCDARP_10_160		2621.35	2779.06	42%	0.03%	0.00%	8.68%
TCDARP_11_160	3222.10	3233.17	3549.53	42%	0.21%	0.00%	4.84%
TCDARP_12_280		7724.32	8533.52	42%		0.27%	8.93%
TCDARP_13_295	6314.24	6329.57	7465.76	47%	-0.08%	0.08%	6.16%
				Avg 41%	0.10%	0.07%	2.79%

Table 11: Economic implications of allowing a later departure of routes (Best solutions on 5 runs)

7 Conclusions

This paper introduces a new variant of the DARP denoted the time consistent DARP. It aims to find trade-off solutions between two objectives: the transportation cost and the time consistency of users. Regarding the literature on the topic, we propose a new formulation of the time consistency of a solution: for a particular user, the number of time classes expresses the number of significantly different service times within one week. We calculate the time consistency of a solution as a lexicographic function of the number of time classes per user. Regarding optimization methods, this formulation is more time consuming than a traditional min-max objective, but it returns a more detailed Pareto front approximation that helps decision makers to select the appropriate solution. TC-DARP extends the TC-VRP by considering time windows and maximum ride times in a problem with multiple destinations. This problem was studied in the context of door-to-door transportation of children with disabilities in region Auvergne-Rhône-Alpes in France.

To compute Pareto front approximations, we developed a matheuristic framework called SPεC based on an epsilon constraint procedure and a set partitioning problem. An initial set of routes is produced by a LNS matheuristic previously proposed for the FSM-DARP-RC in Tellez et al. [37]. Additional routes are generated by an extension of this algorithm to the DARP with multiple time windows and minimal waiting time. At each iteration, a subset of routes is chosen to feed the SPεC procedure. Experiments show the high performance of the SPεC on real-life instances for up to 295 users. SPεC was also been tested on literature instances and was shown to improve the state-of-the-art algorithm for the TC-VRP benchmark.

Some users with disabilities are very sensitive to inconsistent schedules. In this study, we show that economic solutions are already fairly consistent with very few users having 3 time classes. In addition, we found that in most instances, with a small increase in transportation costs (<1%), users schedules with at most 2 time classes are possible. Finally, we show that allowing a flexible departure of routes improves the transportation costs of highly consistent solutions. Future researches concern implementing additional features of the real-life problem such as driver-related constraints (working time, breaks, regulation and driver-consistency) and heterogeneous fleet.

8 Acknowledgements

We would like to thank the European Union through the European Regional Development Fund (ERDF) and the French region Auvergne-Rhône-Alpes for their financial support of the NOMAd project.

A Appendix

A.1 Generation of multiple time windows: $genMTW(\mathcal{H}, u)$

Algorithm 5: Generation of multiple time windows: function $genMTW(\mathcal{H}, u)$

Parameters: u: user considered. $\mathcal{H} = \{h_1, \ldots, h_M\}$: set of the M service times of user u sorted in non-decreasing order.
Data: Λ: width of time classes.
Output: The set of multiple time windows $\mathcal{W}(\mathcal{H}, u)$

```
mtw ← ∅            /* initialize an empty set of multiple time windows */
h̄ ← h_1
h̲ ← h_1
for i = 2, ..., M do
    if h_i > h̲ + Λ then
        mtw ← mtw ∪ {[max{a_{p_u}; h̄ − Λ}, min{b_{p_u}; h̲ + Λ}]}
        h̲ ← h_i
    h̄ ← h_i
mtw ← mtw ∪ {[max{a_{p_u}; h̄ − Λ}, min{b_{p_u}; h̲ + Λ}]}
return mtw
```

A.2 Scheduling algorithm

Algorithm 6: Schedule evaluation

Input: Route $\omega = \{1, ..., M\}$.
Output: The set of service times $h_i \ \forall i \in \omega$ and the maximal route time shift Δ_ω^+, or -1 if **infeasible**

$h_1 \leftarrow a_1$ /* beginning of the service */
$H \leftarrow 0$ /* total waiting time on the route */
$F \leftarrow b_1 - h_1$ /* FTS latest start at node 1 */
$F' \leftarrow b_1 - h_1$ /* FTS earliest start at node 1 */

/* Phase 1: set up nodes at the earliest start */
for $i = 2, \ldots, M$ **do**
  $h_i \leftarrow \max\{a_i; h_{i-1} + \zeta_{i-1} + t_{i-1,i}\}$
  if $h_i > b_i$ **then return** -1
  $H \leftarrow H + \max\{0; a_i - (h_{i-1} + t_{i-1,i} + \zeta_{i-1})\}$
  $F' \leftarrow F$
  $F \leftarrow \min\{F; H + \max\{0; l_i - h_i\}\}$
  if $i = M$ **then** $F' \leftarrow \min\{F'; H\}$

/* Phase 2: optimize route duration */
$\Delta_\omega^+ \leftarrow F - F'$ /* route time shift */
$h_1 \leftarrow h_1 + F'$
for $i = 2, \ldots, M$ **do**
  $h_i \leftarrow \max\{h_{i-1} + \zeta_{i-1} + t_{i-1,i}; a_i\}$

/* Check route duration constraint */
if $(h_M - h_1) > \overline{T}$ **then return** -1

/* Phase 3: check ride time constraints */
for $i = M - 2, \ldots, 1$ **do**
  if $i\mathcal{P}$ **then**
    $u \leftarrow$ user of pickup i /* implies $i = p_u$ */
    $\delta \leftarrow (h_{d_u} - h_{p_u} + \zeta_i) - \overline{T}_u$
    if $(\delta > 0)$ **then** $h_{p_u} \leftarrow h_{p_u} + \delta$
    if $h_{p_u} > b_i$ **then return** -1
    for $j = p_u + 1, \ldots, M$ **do**
      $w_j \leftarrow \max\{a_j; h_{j-1} + \zeta_{j-1} + t_{j-1,k}\}$
      if $h_j > b_j$ **then return** -1
    if $\overline{T}_u - (h_{d_u} - h_{p_u} + \zeta_i) < 0$ **then**
      return -1

return $\{h_i | in \omega\}, \Delta_\omega^+$

A.3 Parameters LSN-SCP

Parameters to generate pool $\mathcal{L}$	
$\chi = 5\%$	record-to-record acceptance criterion.
$penalty = 10000$	penalty cost for incomplete solutions.
$\Phi^- = 10\%$	minimal proportion of removed request used by removal operators.
$\Phi^+ = 45\%$	maximal proportion of removed request used by removal operators.
$p = 6$	roulette wheel parameter for the historical node-pair operator.
$\sigma^+_{init} = 4$-regret	repair operator for building the initial solution
$\eta = 1000$	launch frequency of the SCP.
$Iters = 10000$	max number of iterations.
$\psi = 1.25$	RSCP coefficient to recompute the launch frequency of the SCP.
New parameters to generate pool $\mathcal{L}_{new}$	
$Iters = 250$	max number of iterations.
$\eta = \infty$	the SCP is deactivated.

Table 12: Parameters LNS-SCP [37].

B TC-VRP

Instance	Transportation cost					Time
$C_{\max}$	≤5	≤4	≤3	≤2	≤1	(min)
data5–15	663.2	663.2	663.2	674.1	782.3	11.6
data5–21	773.7	779.1	779.1	779.1	817.3	21.9
data5–25	617.4	617.4	617.4	622.2	669.0	29.1
data5–26	767.6	767.6	771.9	778.8	815.9	45.8
data5–27	934.6	934.6	934.6	942.0	1026.1	61.2
data5–32	984.9	984.9	984.9	989.9	1034.5	22.1
data5–41	1420.7	1420.7	1422.6	1461.6	1615.9	126.9
data5–44	1142.9	1142.9	1142.9	1149.4	1220.3	74.3
data5–46	1458.9	1458.9	1465.4	1492.9	1607.2	114.4
data5–48	1440.8	1449.4	1452.4	1459.8	1597.8	154.9
data5–55	1569.1	1569.1	1571.1	1581.0	1696.9	139.4
data5–59	2714.8	2721.5	2721.5	2883.6	3115.9	281.3
data5–64	2082.0	2082.0	2100.8	2112.4	2304.2	118.9
data5–65	1759.4	1759.4	1766.2	1777.1	1923.9	128.6
data6–15	689.4	689.4	689.4	695.6	741.3	2.2
data6–21	792.0	792.0	796.2	798.1	831.7	40.9
data6–25	680.9	680.9	683.5	683.5	728.7	32.9
data6–26	838.7	838.7	838.7	843.2	905.0	56.0
data6–27	949.8	949.8	949.8	954.0	1060.3	57.9
data6–32	991.1	991.1	991.1	996.2	1013.5	10.4
data6–41	1500.3	1500.3	1504.2	1506.7	1735.5	92.8
data6–44	1239.1	1239.1	1243.9	1244.6	1405.9	76.7
data6–46	1485.1	1485.1	1496.6	1526.0	1597.0	93.0
data6–48	1507.4	1507.4	1519.4	1531.6	1702.8	158.5
data6–55	1816.8	1816.8	1826.3	1854.4	1987.7	121.0
data6–59	2931.3	2931.3	2933.0	3037.4	3389.6	204.1
data6–64	2264.3	2264.3	2271.0	2305.6	2527.0	100.6
data6–65	1981.9	1981.9	1990.6	1998.3	2219.5	163.6

Table 13: Benchmark of [9]

Instance	Transportation cost					Time
$C_{\max}$	≤ 5	≤ 4	≤ 3	≤ 2	≤ 1	(min)
data7–15	746.9	746.9	746.9	752.6	790.3	11.9
data7–21	830.1	833.7	833.7	833.7	899.3	39.8
data7–25	719.5	724.7	724.7	728.8	767.9	34.9
data7–26	880.3	880.3	883.5	887.4	920.7	37.9
data7–27	1053.4	1053.4	1053.4	1056.5	1108.5	39.7
data7–32	1079.7	1079.7	1079.7	1097.3	1154.3	57.2
data7–41	1644.8	1644.8	1648.5	1661.0	1730.4	63.2
data7–44	1295.9	1295.9	1300.3	1307.0	1311.7	20.5
data7–46	1648.6	1648.6	1650.1	1664.6	1715.9	72.4
data7–48	1687.1	1698.9	1698.9	1712.0	1774.3	116.6
data7–55	1889.0	1889.0	1896.3	1918.0	2036.0	136.5
data7–59	3262.4	3262.4	3307.4	3596.6	3892.4	725.2
data7–64	2552.9	2552.9	2552.9	2583.6	2823.4	136.9
data7–65	2196.0	2196.0	2196.0	2224.2	2347.3	159.8
data8–15	773.9	773.9	773.9	780.2	808.5	10.8
data8–21	898.4	898.4	898.4	905.0	956.3	37.8
data8–25	853.3	853.3	853.3	853.3	857.0	0.2
data8–26	962.5	962.5	962.5	970.8	998.2	35.2
data8–27	1184.8	1193.9	1193.9	1193.9	1220.6	17.6
data8–32	1144.0	1144.0	1152.8	1152.8	1181.7	35.8
data8–41	1886.8	1886.8	1888.5	1898.0	1954.3	63.9
data8–44	1409.0	1409.0	1409.0	1414.9	1441.2	28.8
data8–46	1758.5	1758.5	1772.9	1774.8	1817.2	33.8
data8–48	1815.7	1815.7	1820.9	1824.4	1898.1	125.7
data8–55	2007.7	2007.7	2015.4	2037.2	2104.0	120.0
data8–59	3545.7	3545.7	3580.5	3874.4	4020.5	427.4
data8–64	2723.3	2723.3	2723.3	2743.0	2978.2	126.8
data8–65	2404.2	2404.2	2422.9	2433.5	2524.4	159.5
data9–15	797.5	797.5	797.5	804.3	816.1	3.6
data9–21	998.6	998.6	998.6	1003.0	1008.9	5.7
data9–25	894.6	894.6	894.6	894.6	908.1	1.6
data9–26	1024.6	1024.6	1024.6	1024.6	1028.0	0.4
data9–27	1210.6	1210.6	1210.6	1219.5	1241.9	12.3
data9–32	1187.7	1187.7	1199.4	1199.4	1204.9	6.4
data9–41	2022.8	2022.8	2022.8	2022.8	2032.1	6.3
data9–44	1532.5	1532.5	1532.5	1532.5	1595.1	54.1
data9–46	1827.0	1827.0	1827.0	1841.6	1871.0	30.7
data9–48	1973.5	1973.5	1973.5	1992.8	2007.7	28.6
data9–55	2176.0	2176.0	2176.0	2188.1	2275.9	70.3
data9–59	3913.0	3913.0	3916.3	3946.6	4025.3	96.3
data9–64	2942.6	2942.6	2964.9	2964.9	2999.6	28.8
data9–65	2586.5	2586.5	2586.5	2604.7	2624.2	68.0

Table 14: Benchmark of [9]

References

[1] ANAP. *Améliorer la gestion des transports de personnes handicapées*. Français. Tech. rep. Appui santé & médico social, 2016.

[2] Kris Braekers and Attila A. Kovacs. "A multi-period dial-a-ride problem with driver consistency". In: *Transportation Research Part B: Methodological* 94 (Dec. 2016), pp. 355–377.

[3] Stefan Bunte and Natalia Kliewer. "An overview on vehicle scheduling models". In: *Public Transport* 1.4 (Nov. 2009), pp. 299–317.

[4] Samuela Carosi et al. "A matheuristic for integrated timetabling and vehicle scheduling". In: *Transportation Research Part B: Methodological* 127 (Sept. 2019), pp. 99–124.

[5] Vira Chankong and Yacov Y Haimes. *Multiobjective decision making: theory and methodology*. Courier Dover Publications, 2008.

[6] Karl F. Doerner and Juan-José Salazar-González. "Chapter 7: Pickup-and-Delivery Problems for People Transportation". In: *Vehicle Routing*. Ed. by Paolo Toth and Daniele Vigo. Philadelphia, PA: Society for Industrial and Applied Mathematics, Nov. 2014, pp. 193–212.

[7] Michael Drexl. "Synchronization in Vehicle Routing—A Survey of VRPs with Multiple Synchronization Constraints". In: *Transportation Science* 46.3 (Aug. 2012), pp. 297–316.

[8] Alan L. Erera, Martin Savelsbergh, and Emrah Uyar. "Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints". In: *Networks* 54.4 (Dec. 2009), pp. 270–283.

[9] Dominique Feillet et al. "A new consistent vehicle routing problem for the transportation of people with disabilities". In: *Networks* 63.3 (May 2014), pp. 211–224.

[10] Philippe Grangier et al. "A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking". In: *Computers & Operations Research* (2017).

[11] Chris Groër, Bruce Golden, and Edward Wasil. "The Consistent Vehicle Routing Problem". In: *Manufacturing & Service Operations Management* 11.4 (Oct. 2009), pp. 630–643.

[12] Y. Haimes, L. Lasdon, and D. Wismer. "On a bicriterion formulation of the problems of integrated system identification and system optimization". In: *IEEE Transactions on Systems, Man, and Cybernetics* 1 (1971), pp. 296–297.

[13] Sin C. Ho et al. "A survey of dial-a-ride problems: Literature review and recent developments". In: *Transportation Research Part B: Methodological* 111 (2018), pp. 395–421.

[14] Attila A. Kovacs, Sophie N. Parragh, and Richard F. Hartl. "A template-based adaptive large neighborhood search for the consistent vehicle routing problem". In: *Networks* 63.1 (Jan. 2014), pp. 60–81.

[15] Attila A. Kovacs et al. "The Generalized Consistent Vehicle Routing Problem". In: *Transportation Science* 49.4 (Nov. 2015), pp. 796–816.

[16] Attila A. Kovacs et al. "Vehicle routing problems in which consistency considerations are important: A survey". In: *Networks* 64.3 (Oct. 2014), pp. 192–213.

[17] Fabien Lehuédé et al. "A multi-criteria large neighbourhood search for the transportation of disabled people". In: *Journal of the Operational Research Society* 65.7 (2014), pp. 983–1000.

[18] Hongtao Lei, Gilbert Laporte, and Bo Guo. "Districting for routing with stochastic customers". In: *EURO Journal on Transportation and Logistics* 1.1-2 (2012), pp. 67–85.

[19] Zhixing Luo et al. "On service consistency in multi-period vehicle routing". In: *European Journal of Operational Research* 243.3 (June 2015), pp. 731–744.

[20] Dennis Luxen and Christian Vetter. "Real-time routing with OpenStreetMap data". In: *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. GIS'11. Chicago, Illinois: ACM, 2011, pp. 513–516.

[21] Mathias Michaelis and Anita Schöbel. "Integrating line planning, timetabling, and vehicle scheduling: a customer-oriented heuristic". In: *Public Transport* 1.3 (Aug. 2009), p. 211.

[22] Ashlea Bennett Milburn and Jessica Spicer. "Multi-objective home health nurse routing with remote monitoring devices". In: *International Journal of Planning and Scheduling* 1.4 (2013), pp. 242–263.

[23] Yves Molenbruch, Kris Braekers, and An Caris. "Typology and literature review for dial-a-ride problems". In: *Annals of Operations Research* 259.1-2 (Dec. 2017), pp. 295–325.

[24] Wlodzimierz Ogryczak et al. “Fair optimization and networks: A survey”. In: *Journal of Applied Mathematics* 2014 (2014).

[25] Junhyuk Park and Byung-In Kim. “The school bus routing problem: A review”. In: *European Journal of Operational Research* 202.2 (Apr. 2010), pp. 311–319.

[26] David Pisinger and Stefan Ropke. “A general heuristic for vehicle routing problems”. en. In: *Computers & Operations Research* 34.8 (2007), pp. 2403–2435. ISSN: 03050548.

[27] Ulrike Ritzinger, Jakob Puchinger, and Richard F Hartl. “A survey on dynamic and stochastic vehicle routing problems”. In: *International Journal of Production Research* 54.1 (2016), pp. 215–231.

[28] Verena Schmid and Jan Fabian Ehmke. “Integrated timetabling and vehicle scheduling with balanced departure times”. In: *OR Spectrum* 37.4 (Oct. 2015), pp. 903–928.

[29] Michael Schneider et al. “Territory-Based Vehicle Routing in the Presence of Time-Window Constraints”. In: *Transportation Science* 49.4 (Nov. 2015), pp. 732–751.

[30] Karen Smilowitz, Maciek Nowak, and Tingting Jiang. “Workforce Management in Periodic Delivery Operations”. In: *Transportation Science* 47.2 (May 2013), pp. 214–230.

[31] Remy Spliet and Guy Desaulniers. “The discrete time window assignment vehicle routing problem”. In: *European Journal of Operational Research* 244.2 (2015), pp. 379–391.

[32] Remy Spliet and Adriana F. Gabor. “The Time Window Assignment Vehicle Routing Problem”. en. In: *Transportation Science* 49.4 (Nov. 2015), pp. 721–731.

[33] Anirudh Subramanyam and Chrysanthos E Gounaris. “A branch-and-cut framework for the consistent traveling salesman problem”. In: *European Journal of Operational Research* 248.2 (2016), pp. 384–395.

[34] Ilgaz Sungur et al. “A Model and Algorithm for the Courier Delivery Problem with Uncertainty”. In: *Transportation Science* 44.2 (May 2010), pp. 193–205.

[35] C.D. Tarantilis, F. Stavropoulou, and P.P. Repoussis. “A template-based Tabu Search algorithm for the Consistent Vehicle Routing Problem”. In: *Expert Systems with Applications* 39.4 (Mar. 2012), pp. 4233–4239.

[36] Oscar A. Téllez Sánchez. “Optimisation du transport quotidien des personnes en situation de handicap”. PhD thesis. INSA Lyon, Sept. 2019.

[37] Oscar Tellez et al. “The fleet size and mix dial-a-ride problem with reconfigurable vehicle capacity”. In: *Transportation Research Part C: Emerging Technologies* 91 (2018), pp. 99–123.

[38] Paolo Toth and Daniele Vigo. “Heuristic algorithms for the handicapped persons transportation problem”. In: *Transportation Science* 31.1 (1997), pp. 60–71.

[39] Zefeng Xu and Yanguang Cai. “Variable neighborhood search for consistent vehicle routing problem”. In: *Expert Systems with Applications* 113 (Dec. 2018), pp. 66–76. ISSN: 09574174.

[40] Hongsheng Zhong, Randolph W. Hall, and Maged Dessouky. “Territory Planning and Vehicle Dispatching with Driver Learning”. In: *Transportation Science* 41.1 (Feb. 2007), pp. 74–89.