The time-consistent dial-a-ride problem

Oscar Tellez, Samuel 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

January 16, 2020

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 users 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 users 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. Most passengers submit the same transportation request everyday but variations are common (attendance, pickup or destination modifications according to medical appointments, etc.). In 2016, we carried out a statistical study on field data which shows that only 30 % of passengers had a complete and regular schedule throughout the week. A passenger demand variation may impact the schedule of other passengers on the same route. As a result, passengers and and MSIs express a demand for regularity (or consistency) in service times. For medical, cognitive or convenience reasons, it is desirable for a passenger who has the

same need on several week-days to haves the same pickup / drop off time. In this paper we seek to design an algorithm to be integrated into a dial-a-ride application for passengers with regularity requirements.

This work has been motivated by a real-life case study in the area of Lyon, France. Transportation of disabled or elderly people in the area is operated mainly by a single carrier[1] who works for multiple MSIs and has a fleet of adapted vehicles. Every morning, from Monday to Friday, disabled children from the region are transported from their home to a MSI. In the afternoon, they are driven back home. Without a loss of generality, this paper presents the results of our research for morning trips. Thus, 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). This research has been conducted in close cooperation with SMIs and the carrier company.

The paper is organized as follows: Section 2 presents how the TC-DARP is related to the existing literature in operations research. In Section 3, we give a formal definition of the TC-DARP and formulate it as a mixed-integer linear program (MILP). Section 4 presents a general approach for solving the TC-DARP. Section 5 details the algorithm used for generating the routes. In Section 6, computational results and management insights are reported.

2 Related literature

The mono-period DARP of our application has been presented in Tellez et al. [29]. We focus our literature review 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 [8] and were rapidly extended to passenger transportation [6]. Readers interested in an extensive review on vehicle routing with consistency considerations can refer to Kovacs et al. [13]. 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 [23, 18]. In Feillet et al. [6], 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 [15, 32, 23, 22].

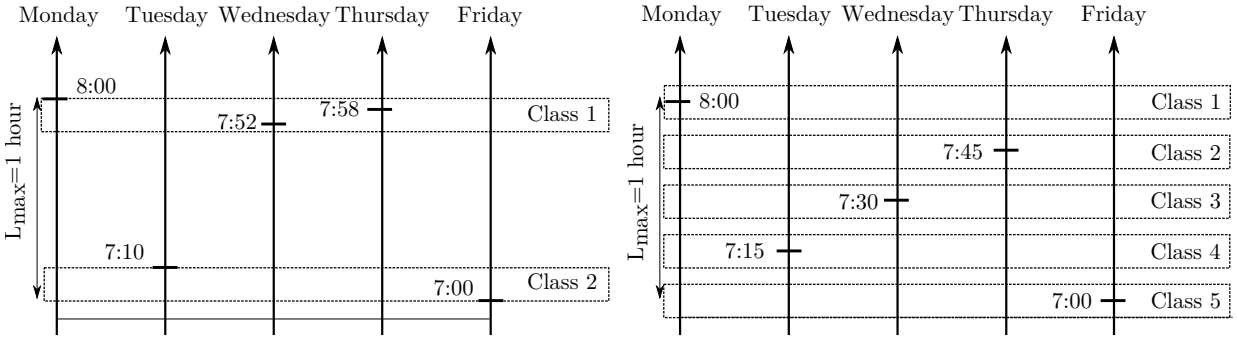
This paper focuses on service time-consistency applied to a Dial-a-Ride Problem (DARP). In contrast to the VRP, the DARP considers one origin and destination for each user and a maximum ride time. The main applications of the DARP concern door-to-door transportation of people, particularly elderly or disabled people [10, 30, 14].

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. [8] 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

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



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

(b) Solution with $C_{\max} = 5$ 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.

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. [6] 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 users, 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. Regularity is then improved by minimizing the maximum number $C_{\max}$ of time-classes over all users. 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), 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. In Figure 1(b), as service times are evenly spread between 7:00 and 8:00, there are 5 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 users who have $C_{\max}$ time-classes is equivalent to another solution where only 1% of users 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 $C_{\max} - 1$, $C_{\max} - 2$ and so forth.

To the best of our knowledge, this approach is a new refinement of the Feillet et al. [6] 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 [19], 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. [8], 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. [27, 28, 11, 12, 16, 31]). The current best results on the Groër et al. [8] benchmark instance set was obtained by Xu and Cai [31], 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. [12] 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 [26] 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. [6] 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 approach is the Time Window Assignment Vehicle Routing Problem (TWAVRP) introduced by Spliet and Gabor [25]. 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 traveling cost. The TWAVRP is a particular case of the genConVRP if scenarios are seen as periods and the number of drivers per user 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 [24] propose a variant, called the discrete time window assignment vehicle routing problem, where the chosen time windows are selected from a discrete set.

Consistency issues are also often related to having stochastic customers in the VRP [21]. For example, Sungur et al. [27] 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. [5] 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 [4].

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 use the notion of time-classes introduced by Feillet et al. [6] and explore solutions where some users accept several time-classes. Compared to the TCVRP proposed in Feillet et al. [6], we propose a refinement of the $C_{\max}$ minimization approach.

Regarding the VRP attributes, we investigate a more enriched setting than Feillet et al. [6]. In particular, we consider time windows and maximum ride times in a problem with multiple MSIs (e.g. multiple pickup and delivery locations). As a result, solutions might contain routes that contain waiting times. We keep the assumption that improving consistency by artificially introducing waiting times within routes is not realistic with respect to drivers practice. Similarly to Kovacs et al. [12], we consider that the departure time of routes can be changed in order to improve consistency.

As we consider a para-transit user application, we will refer to the passengers of the transportation system as *users* in the remainder of the paper.

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

We consider a set of users $\mathcal{U}$ to be transported during a planning horizon $\mathcal{T}$. Each uses may require a particular space $v \in \mathcal{V}$ in the vehicle such as a seat or wheelchair space. There is a homogeneous fleet of vehicles based at a single depot o. The vehicle capacity is defined by a vector $\mathcal{Q} = \{Q_1, \ldots, Q_{|\mathcal{V}|}\}$ representing the availability of each space type $v \in \mathcal{V}$.

Each user $u \in \mathcal{U}$ has a pickup node denoted by $p_u \in \mathcal{P}$, a delivery node $d_u \in \mathcal{D}$, a maximum ride time $\overline{T}_u$, and a demand indicator $\beta_u^t \in \{0, 1\}$ for each period $t \in \mathcal{T}$. $\beta_u^t = 1$ if user u requires transportation at period t and 0 otherwise. Each user can be serviced at most once in any period. Note that if two users have a common origin or destination, nodes are duplicated so that each pickup node and each delivery node has exactly one user.

The TC-DARP is defined on a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with the vertex set $\mathcal{V} = \{\mathcal{P} \cup \mathcal{D} \cup o\}$ and the arcs set $\mathcal{A}$ 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 $i \in \mathcal{V}$ is associated with a service duration ζ_i and a time window $[a_i, b_i]$. Every

$\mathcal{U}$	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
Ω	set of all routes
Ω_u	set of routes serving user $u \in \mathcal{U}$
$\mathcal{C}$	set of time-classes

Table 1: 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
C_ω	variable cost of route $\omega \in \Omega$
λ	weekly vehicle cost ownership 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 2: Data

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}.

We propose a route-based MILP formulation of the TC-DARP. A route is any feasible sequence of nodes visited by a vehicle. Each route ω starts and finishes at node o and is characterized by a set of visited pickup and delivery nodes. Every pickup or delivery node $i \in \mathcal{P} \cup \mathcal{D}$ on a route ω is 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 and satisfies time windows and maximum ride time constraints for each user on the route. In addition, for each route a *maximum route time shift* Δ_ω^+ which represents the maximal amount of time by which its departure time can be postponed, without violating any time window or ride time constraints. The detailed scheduling procedure for the calculation of values Δ_ω^+ is presented in Appendix A.2.

Each route $\omega \in \Omega$ is operated by one vehicle. To each vehicle is associated: a weekly ownership cost λ, a cost per kilometer τ related the fuel consumption, and a cost per hour α related to the driver's cost. The TC-DARP can be seen as 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 to minimize the sum of fixed and variable traveling costs. The second objective is to minimize the service time inconsistency.

Note that we do not explicitly consider users with more than one address. Actually, time-consistency is meaningful for a given address. A person with two distinct addresses is modeled as two different people (one per address).

Tables 1, 2 and 3 synthesize the mathematical notations for the sets, data and variables used in the TC-DARP mathematical model.

To model the problem, we introduce the binary decision variable y_ω^t which is equal to 1 if route $\omega \in \Omega$ is selected at period $t \in \mathcal{T}$. The binary variable z_{uc}^t is equal to 1 if user $u \in \mathcal{U}$ is assigned to time-class $c \in \mathcal{C}$ at period $t \in \mathcal{T}$. Binary variables μ_{uc} indicate which time-classes from set $\mathcal{C}$ are actually used by user u. If u has 3 time-classes, we assume that time-classes $1, 2$ and 3 are used, and time-classes 4 and 5 are not used. Hence, $\mu_{u,1} = \mu_{u,2} = \mu_{u,3} = 1$. The shift of route $\omega \in \Omega$ departure time at period $t \in \mathcal{T}$ is expressed by continuous variables $\delta_\omega^t \in [0, \Delta_\omega^+]$.

Objectives

The first objective is the minimization of transportation costs that is defined by the sum of fixed and routing costs. Fixed costs are related to the cost of owning the vehicles. Then, the cost C_ω of a route $\omega \in \Omega$ will then depend on its duration (which may include some waiting time) and on its length.

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

The second objective minimizes time inconsistency, which is modeled with a lexicographical refinement of the time-class model proposed by Feillet et al. [6]. The expression used in the MILP model is the

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 $t \in \mathcal{T}$
v	number of vehicles needed for the whole planning horizon
m_u	number of users having $c \in \mathcal{C}$ time-classes (post-processed variable)

Table 3: Variables

following:

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

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

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

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:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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 presents the first component of the framework, which finds the initial solution as well as an initial pool of routes. Section 4.3 details the second key component, which is a mono-objective optimization procedure for the TC-DARP. Then, Section 4.4 presents the rules to select an appropriate subset of routes at each step of the algorithm.

4.1 The SPεC matheuristic framework

The general framework, introduced in Algorithm 1, is based on an ε-constraint procedure [9, 3]. 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. 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 4.2 (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.

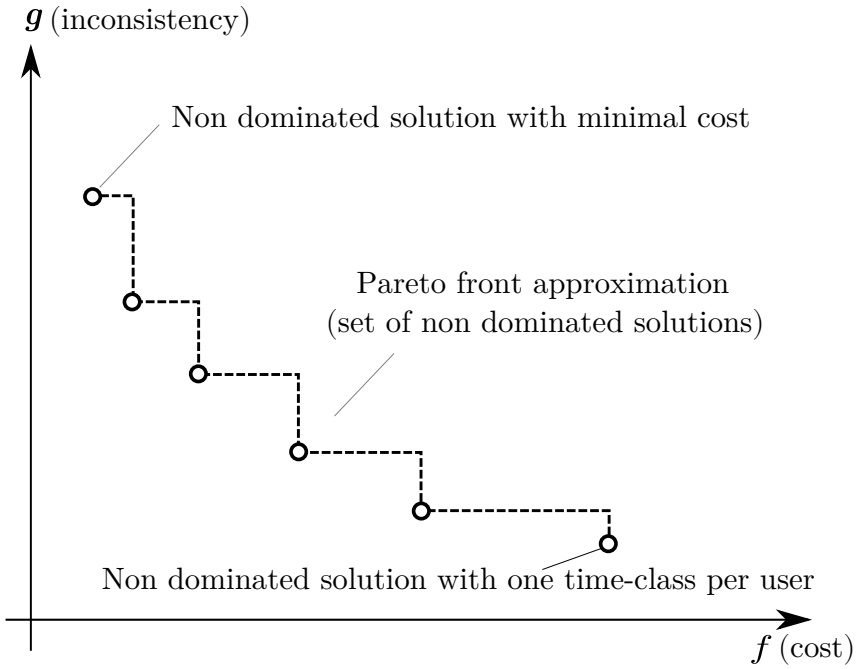


Figure 2: Pareto front approximation designed by the SPεC algorithm

Algorithm 1: The SPεC framework

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

`/* Initialization */`

$ParetoFront \leftarrow \emptyset$
$\mathcal{L} \leftarrow \emptyset$: Pool of routes
$S \leftarrow \emptyset$: temporary solution
$(S^*, \mathcal{L}) \leftarrow initialize()$ `/* See Section 4.2 */`
$\bar{f} \leftarrow f(S^*)$: cost upper limit

`/* Iterations */`

while *stopping criterion is not met* **do**

`/* Optimize inconsistency and cost objectives, see Section 4.3 */`

$(S, \mathcal{L}) \leftarrow solveMonoTCDARP(\text{lexmin}\, \boldsymbol{g}, f \leq \bar{f}, S^*, \mathcal{L})$
$(S, \mathcal{L}) \leftarrow solveMonoTCDARP(\min f, \boldsymbol{g} \leq_{lex} \boldsymbol{g}(S), S, \mathcal{L})$

`/* Update solution */`

if $(f(S) < f(S^*)) \vee (\boldsymbol{g}(S) <_{lex} \boldsymbol{g}(S^*))$ **then**
  $S^* \leftarrow S$
  Update $ParetoFront$ with solution S^*
else
  $\varepsilon \leftarrow \phi \times \varepsilon$
end

`/* End of one iteration */`

Update epsilon constraint: $\bar{f} \leftarrow f(S^*) \times (1 + \varepsilon)$
Limit the size the $\mathcal{L}$ to N_{max}

end
return $ParetoFront$

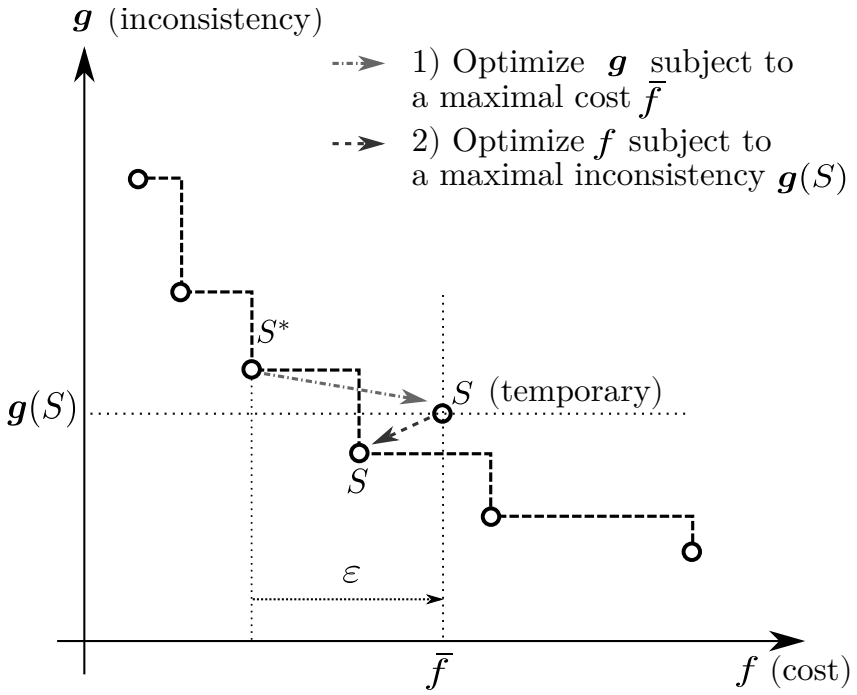


Figure 3: Pareto front exploration with the two optimization procedures of one iteration (Algorithm 1 lines 7 and 8).

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 step 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 by the *consistency-first* criteria described in Section 4.4 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 $\boldsymbol{g} = (0, \ldots, |\mathcal{U}|)$.

4.2 Initialization: cost minimization

The set of non-dominated solutions of the TC-DARP defines a Pareto front in which one of the extreme points corresponds to a solution with minimum cost and possibly with a high number of time-classes. This extreme point can be found by solving a simplified version of the TC-DARP, called multi-period DARP (MP-DARP) that ignores the consistency requirements. This MP-DARP is modeled as follows:

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

$$s.t.$$

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

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

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

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

The objective function (20) is the same as Equation (1). It represents the transportation costs. Constraints (21) are set covering constraints for demand satisfaction. Constraints (22) enforce the number

of vehicles to be less than or equal to v at each period. This problem is a set covering problem that is easily solved to optimality by a solver, provided the number of routes remains reasonable. Moreover, MP-DARP solutions provide feasible routes for the TC-DARP. Hence, the MP-DARP model can be used to build a good initial solution to the TC-DARP.

In order to define a non-dominated route, we define route dominance as follows:

Definition 4.1. Route dominance. Let us consider two routes $\omega \in \Omega$ and $\omega' \in \Omega$, with respective costs C_ω and $C_{\omega'}$. Route ω is said to *dominate* ω' if both routes visit the same set of users (in any order) and $C_\omega \leq C_{\omega'}$. A route is said to be non-dominated in a set if there is no other route in this set that dominates it.

Definition 4.2. 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 4.2 is illustrated by Figure 4. 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.

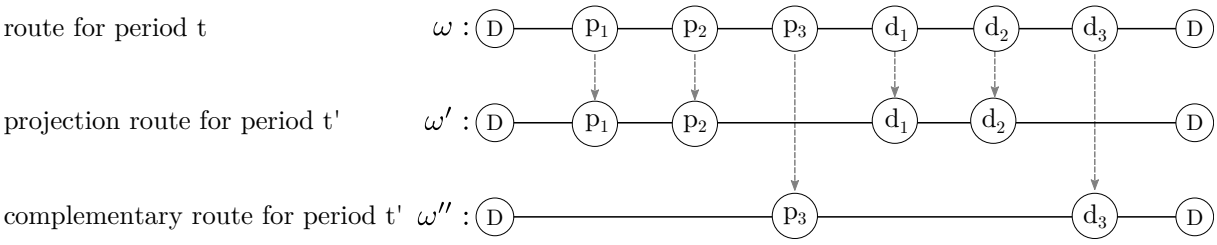


Figure 4: Example of projection and complementary routes

Algorithm 2 presents the MP-DARP algorithm. A key element of this algorithm is the pool of routes $\mathcal{L}$, that is initialized with the routes found while solving one independent DARP for each period $t \in \mathcal{T}$ (line 4). Each independent DARP minimizes the total transportation cost composed by the distance-related cost and the time-related cost for a specific period t. In our case, we use the LNS-SCP matheuristic proposed by Tellez et al. [29]. It is a large neighborhood search-based procedure that quickly yields feasible routes with minimal duration and optimized with respect to the cost.

During the execution of the LNS-SCP algorithm, every time a new solution is accepted as the current solution, all the routes of this solution are saved in the pool $\mathcal{L}$. As we can see in the model of MP-DARP, the order of the nodes and the service times of the routes are not considered. Thus, this problem can be solved by using a subset of $\mathcal{L}$ considering only non-dominated routes (see Definition 4.1). Note that, we still save all the routes of every new current solution in $\mathcal{L}$, even if they are dominated, because dominated routes can be useful to solve the TC-DARP.

Each iteration of Algorithm 2 consists of four steps:

In step 1, a subset $l \subset \mathcal{L}$ of N^{init} non-dominated routes is selected with some selection rule $r \in \mathcal{R}$ (line 11). The selection rules will be described in Section 4.4. The set l is enriched with the projection and complement of its routes (line 12) and then added to the restricted pool of routes $\mathcal{L}'$ (line 13).

In step 2, an MP-DARP instance is defined in the pool $\mathcal{L}'$. It is solved by an MILP solver with a time limit t_{max} (line 14). If the best solution S^* is not empty, it is used as a warm start to initialize the MILP solver, which considerably reduces computation time.

Given that the formulation of the MP-DARP is a set covering problem and not a set partitioning problem, 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 supposed). This is done (line 16) by solving the MP-DARP again with another pool of routes and a warm-start on S. The other pool of routes is initialized with the routes of the solution S and enriched as follows. For each user u visited more than once and for each route ω that visits user u, a new route ω' identical to route ω but that does not visit user u is added to the pool. All the new routes constructed during the repair of the solution (line 16) are also added to $\mathcal{L}$ (line 17).

Step 3 performs pool management (lines 19-21). The current pool of routes is cleared if the MP-DARP could not be solved to proven optimality. This mechanism keeps a manageable pool size and is inspired by previous research [7, 29]. Step 4 updates the current best solution S^* and the counter of iterations

Algorithm 2: initialize()

Parameters: t_{max}: solver time limit, N^{init}: number of selected routes, $\mathcal{R}$: list of selection rules, $MaxIter$: maximal number of iterations without improvement

$\mathcal{L} \leftarrow \emptyset$: Pool of routes
for *each period* $t \in \mathcal{T}$ **do**
  Solve a DARP for period t with algorithm LNS-SCP [29]
  Add all routes generated to $\mathcal{L}$
end
$S \leftarrow \emptyset$: current solution
$S^* \leftarrow \emptyset$: best solution found
$\mathcal{L}' \leftarrow \emptyset$: restricted pool of routes
$itNonImp \leftarrow 0$: number of iterations without improvement of S^*
while $itNonImp < MaxIter$ **do**
  `/* 1. Select routes */`
  Select a subset $l \subseteq \mathcal{L}$ of N^{init} non-dominated routes using a selection rule $r \in \mathcal{R}$
  Enrich l with the projection and complement of its routes
  Add routes l into $\mathcal{L}'$
  `/* 2. Solve the MP-DARP */`
  Compute S by solving an MP-DARP with time limit t_{max}, pool $\mathcal{L}'$ and warm start on S^*
  if *at least one user is visited more than once on a period* **then**
    Repair S such that each user is visited exactly once
    Add to $\mathcal{L}$ all new routes created during the reparation
  end
  `/* 3. Pool management */`
  if *MP-DARP is not solved to proven optimality* **then**
    $\mathcal{L}' \leftarrow \emptyset$;
  end
  `/* 4. Update best solution and` $itNonImp$ `*/`
  if $f(S) < f(S^*)$ **then**
    $S^* \leftarrow S$
    $itNonImp \leftarrow 0$
  else
    $itNonImp \leftarrow itNonImp + 1$
  end
end
return $S^*, \mathcal{L}$

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. This is why, the number $itNonImp$ of iterations without any improvement is used as a stopping criterion.

4.3 Mono-objective optimization procedure

This section presents the procedure to solve the TC-DARP with one objective with a constraint on the maximal value of the other objective. This procedure, detailed in Algorithm 3, is called by Algorithm 1, either to minimize inconsistency subject to a maximal cost constraint (line 7) , or to minimize cost subject to a maximal inconsistency constraint (line 8) .

Algorithm 3: solveMonoTCDARP(z, $\mathcal{E}$, S_{ini}, $\mathcal{L}$)

Arguments: z: objective, $\mathcal{E}$: epsilon constraint, S_{ini}: initial solution, $\mathcal{L}$: pool of routes.
Parameters: $t_{\max}$: solver time limit, N : number of routes to append to the pool $\mathcal{L}$ at each iteration, $MaxIter$: maximum number of iterations without improvement, $\mathcal{O}$: list of policies.
Result: best solution found S^*

1 $S \leftarrow \emptyset$: current solution
2 $S^* \leftarrow S_{ini}$: best solution found
3 $\mathcal{L}' \leftarrow \emptyset$: restricted pool of routes
4 $itNonImp \leftarrow 0$: number of iterations without improving S^*
5 **while** $itNonImp < MaxIter$ **do**
  /* 1. Select routes */
6 Select a policy composed by a route source s and a selection rule r from the list $\mathcal{O}$
7 **if** *the route source s is the pool $\mathcal{L}$* **then**
8 Select a subset $l \subseteq \mathcal{L}$ of N routes using r /* see Section 4.4 */
9 **else**
10 Generate new routes by solving a DARP with multiple time windows:
  $\mathcal{L}_{new} \leftarrow solveDARPmTW(S^*)$ /* See Section 5 */
11 Save new routes: $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{L}_{new}$
12 Select a subset $l \subseteq \mathcal{L}_{new}$ of N routes using r /* see Section 4.4 */
13 **end**
14 Enrich l computing for each route the projection and complementary routes
15 $\mathcal{L}' \leftarrow \mathcal{L}' \cup l$
  /* 2. Solve a modified TC-DARP model. */
16 Compute solution S by solving model (5–18) with objective z, epsilon constraint $\mathcal{E}$, on route pool $\Omega = \mathcal{L}'$, time limit $t_{\max}$ and initial solution S^*
  /* 3. Pool management */
17 **if** *TC-DARP is not solved to proven optimality* **then**
18 $\mathcal{L} = \{\}$
19 **end**
  /* 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 **end**
26 **end**
27 **return** $S^*, \mathcal{L}$

Algorithm 3 uses the same structure as Algorithm 2. However, it solves a TC-DARP instead of an

MP-DARP.

The arguments of Algorithm 3 are the objective function z to be minimized, epsilon constraint $\mathcal{E}$ which defines the upper bound on the previous objective, the initial solution S_{ini}, and the pool of routes $\mathcal{L}$. The main variables of the algorithm are the restricted pool of routes $\mathcal{L}'$ (initially empty), the current solution S, and the best found solution S^* which is initialized to the solution S_{ini} (lines 1-4).

Each iteration of Algorithm 3 consists in four steps: 1) selecting routes; 2) solving the mono-objective TC-DARP given a 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 (see line 5). Steps 3 and 4 are the same as in Algorithm 2. We therefore detail here steps 1 and 2 only.

Again, a key element of this algorithm is the pool of routes $\mathcal{L}$. This pool has been initialized by Algorithm 2. In Step 1, the set of selected routes is chosen by using a policy defined by one source of routes s and one selection rule r. The source s provides a pool of routes from which a selection rule r chooses a subset l. There are two distinct sources of routes. The first one is the pool of routes $\mathcal{L}$ that was generated in Algorithm 2 (lines 7–8). The second source of routes is a pool of new routes denoted $\mathcal{L}_{new}$ (lines 9–12). New routes are created by solving a Dial a Ride Problem with multiple Time Windows (DARPmTW). The construction and the solving of the DARPmTW will be presented in Section 5. The new routes in $\mathcal{L}_{new}$ are added to $\mathcal{L}$ (line 11) to enable their selection in further iterations.

Whatever the sources, a subset l of N routes is selected in the source using the selection rule r (line 8 and 12). Multiple selection rules have been designed. Each selection rule proposes a different way of choosing routes from a source according to a given criterion. A detailed description of each rule is presented in Section 4.4.

For diversification purposes, we define several policies combining the sources and the selection rules into a list $\mathcal{O} = \{(s_1, r_1), \ldots, (s_{|\mathcal{O}|}, r_{|\mathcal{O}|})\}$. Policies are selected sequentially with the following rule: if solution S^* has not been improved at the current iteration, the next policy is selected. Otherwise, the same policy is kept at the next iteration. When the last policy in $\mathcal{O}$ is reached, the procedure starts again from the first policy.

The selected routes as well as their projection and complementary routes are added to the current pool $\mathcal{L}'$ (lines 14 and 15).

In step 2, line 16, an MILP is solved. This problem optimizes objective z subject to constraints (5–18), presented in Section 3, and subject to epsilon constraint $\mathcal{E}$. The current pool $\mathcal{L}'$ is used ($\Omega = \mathcal{L}'$). Then, the MILP solver time limit is set at $t_{\max}$ and the solution S^* is used as initial solution.

There are two cases, depending on whether Algorithm 3 is called at line 7 or 8 of Algorithm 1. In the first case, the objective z is to minimize the inconsistency and the epsilon constraint limits the maximal cost. In second case, the objective z is to minimize the cost and the lexicographic epsilon constraint limits the maximal inconsistency.

4.4 Selection rules

When the restricted pool $\mathcal{L}'$ becomes too large, the MILP solver cannot improve the given initial solution. Hence, a key element is to select a subset of routes of reasonable size in $\mathcal{L}$ (line 8) or $\mathcal{L}_{new}$ (line 12). We present in this section three performance indicators used to guide route selection and two route selection strategies called Sequential Selection and Random Biased Selection.

4.4.1 Performance indicators

The route selection algorithms are based on the following performance indicators which aim at evaluating if a route $\omega \in \mathcal{L}$ should be integrated in the TC-DARP route pool.

- **Solution cost F_ω:** This is the cheapest solution cost among all TC-DARP solutions found so far that uses route ω on at least one period. Initially, F_ω is set to infinity.
- **Solution inconsistency G_ω:** This is the inconsistency of the least inconsistent solution found among all TC-DARP solutions found so far that uses route ω in at least one period. Initially, G_ω is set to infinity.
- **Sub-problem cost J_ω:** This is the cost of the cheapest one-period solution containing route ω.

Two sorting criteria are defined. The first sorting criterion is the cost-first criterion $(F_\omega, J_\omega, C_\omega)$ that lexicographically sorts the routes in ascending order, first by the solution cost F_ω, then by the sub-problem

cost J_ω and finally by the route routing cost C_ω. The second sorting criterion is the inconsistency-first criterion $(G_\omega, F_\omega, J_\omega)$ that lexicographically sorts the routes in ascending order, first by the solution inconsistency G_ω, then by the solution cost F_ω, and finally by the sub-problem cost J_ω. This last criterion is also used to limit the size of the pool $\mathcal{L}$ in line 16 of Algorithm 1.

4.4.2 Random Biased Selection (RBS)

The Random Biased Selection rule is inspired by the randomized selection mechanisms used in Pisinger and Ropke [20]. The pool $\mathcal{L}$ is first sorted according to the lexicographic order defined by criterion $(F_\omega, J_\omega, C_\omega)$ when the objective is to minimize costs, and criterion $(G_\omega, C^t_\omega, C_\omega)$ when the objective is to minimize time-inconsistency. N routes are then chosen based in their position in the sorted list $\mathcal{L}$, according to the following rule: the route at position $\xi^\rho \times |\mathcal{L}|$ is selected, where $0 \leq \xi < 1$ is a random number and $\rho > 1$. This route is then removed from the list and the process repeats until N routes are selected. Because the list is sorted in decreasing order of the performance indicator, this mechanism gives a higher selection probability to the routes with better performance.

4.4.3 Sequential Selection (SS)

The purpose of the Sequential Selection (SS) rule is to browse the pool of routes $\mathcal{L}$ in decreasing order of their performance indicator, selecting a significant number of new routes at each iteration and allowing for the selection again of a few of the previously selected routes.

This rule is based on the same sorting principle as the RBS rule: depending on the considered objective function, routes are first sorted according to either a cost-first criterion $(F_\omega, J_\omega, C_\omega)$ or an inconsistency-first criterion $(G_\omega, J_\omega, C_\omega)$. From the sorted list, routes are processed one by one until N routes are selected. For each route, we check if it was selected in a previous call or not. If it was previously selected, it can be selected again with a probability γ. Otherwise, it is always selected. Figure 5 shows an example of SS with $N = 6$.

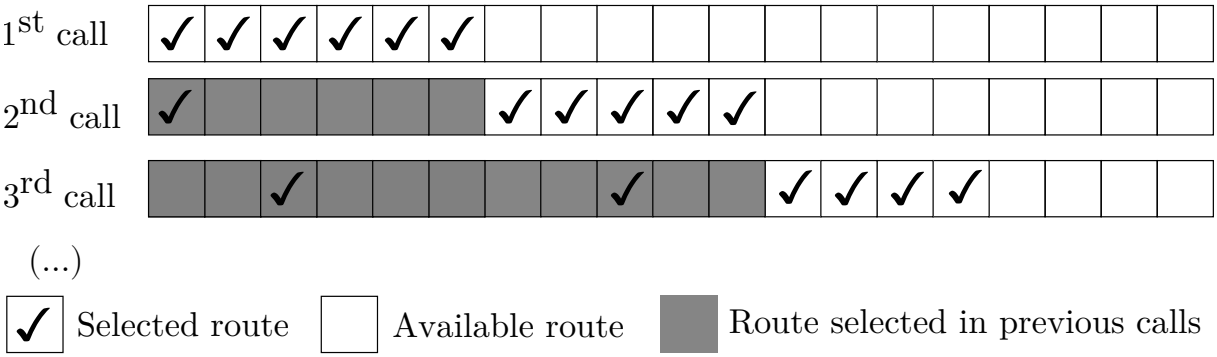


Figure 5: Example of 3 successive calls to the sequential selection rule with $N = 6$.

The sequential selection mechanism increases the probability of selecting together routes that have been part of the same solution with the same overall performance (F_ω or G_ω). The routes must be sorted each time the sequential selection is called, as performance indicators such as F_ω and G_ω may change from one iteration to another.

The record of previously selected routes is re-initialized every time the solveMonoTCDARP (Algorithm 3) is called.

5 New routes generation: the DARP with multiple time windows

The SPεC framework is initialized with routes that have been generated by solving independent DARPs. These routes are combined in order to form more consistent solutions. Time-consistency can nevertheless be improved by generating routes that are dominated in terms of costs (i.e. routes of which the cost can be improved by modifying the sequence of visits).

These dominated routes, used in Algorithm 3 (line 10), may never be generated by the LNS operators used to solve the DARP. This section describes the procedure used to generate new routes that are likely to improve time-consistency. The proposed approach was inspired by Feillet et al. [6].

Section 5.1 presents the main process for the generation of a new set of routes denoted $\mathcal{L}_{new}$. Section 5.2 shows how time windows are defined.

5.1 Main process for the generation of new routes

The process for generating new routes consistent with the current solution works as follows:

1. One user $\bar{u}$ is selected randomly among the set of users that have the maximum number of time-classes $C_{\max}$.
2. For each period $t \in \mathcal{T}$, new routes are generated as follows:
   (i) A set $\mathcal{W}_u^t$ of multiple time windows are defined for each user $u \in \mathcal{U}$ according to their service times: if $u = \bar{u}$, multiple time windows are defined in order to decrease its current number of time-classes, while for the rest of users ($u \neq \bar{u}$) the number of time-classes can be maintained (Section 5.2).
   (ii) Routes are generated by solving an auxiliary Dial-A-Ride Problem with multiple Time Windows (DARPmTW) for period t. The routes generated during this process are gathered into a sub-pool of routes denoted $\mathcal{L}_{new}^t$ (Section 5.3).
3. The sub-pools $\mathcal{L}_{new}^t$ are gathered into a single pool denoted $\mathcal{L}_{new} = \cup_{t\in\mathcal{T}}\mathcal{L}_{new}^t$ and returned to SPεC (line 10, Algorithm 3).

In Feillet et al. [6], the TC-VRP does not have time windows. The auxiliary problem solved is a vehicle routing problem with multiple time windows and no waiting time (VRPmTW-nw). The multiple time windows help to define new routes that can decrease the number of time-classes for user $\bar{u}$. Waiting times are forbidden since the insertion of unnecessary waiting times could artificially improve consistency.

In our case, the initial problem integrates time windows. The sub-problem is a Dial-A-Ride Problem with multiple Time Windows and minimal route duration (DARPmTW). Routes may integrate waiting times due to time windows but waiting times that artificially improve time-consistency are still forbidden.

5.2 Definition of time windows for the DARPmTW

Let $\mathcal{W}_u^t$ be the set of multiple time windows to be defined for each user $u \in \mathcal{U}$ in the DARPmTW solved to generate new routes for period t. If $u = \bar{u}$, $\mathcal{W}_u^t$ is defined to decrease its current number of time-classes, while for the rest of users ($u \neq \bar{u}$) $\mathcal{W}_u^t$ is defined such that the number of time-classes can be maintained.

Let us recall some of the notations: we consider a user $u \in \mathcal{U}$ with a pickup time window $[a_{p_u}, b_{p_u}]$, the subset of time periods with transportation demands $\mathcal{T}_u \subseteq \mathcal{T}$ and variables h_u^t representing the pickup service time of user u at period $t \in \mathcal{T}_u$. We denote by $\mu(\mathcal{H})$ the number of time-classes for the set of service times $\mathcal{H}$. So $\mu(\mathcal{H}_u)$ is the number of time-classes of user u. The service times are allocated to time-classes as in Feillet et al. [6].

In the formal definition of $\mathcal{W}_u^t$, we use a function $\mathcal{W}(\mathcal{H}, u)$. This function returns a set of time windows within $[a_{p_u}, b_{p_u}]$. Any service time inserted in these time windows will not create a new class. This is illustrated on Figure 6. The detailed construction process is given in Appendix A.1.

For each user $u \in \mathcal{U}$, the answers to the following two *yes/no* questions determine how the multiple time windows $\mathcal{W}_u^t$ are defined:

Q1 Is u the selected user $\bar{u}$?

Q2 Is the number of time-classes of user u decreased by 1 if 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 answers to questions **Q1** and **Q2** yield four ways to define multiple time windows:

Yes/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. So $\mathcal{W}_{\bar{u}}^t = \mathcal{W}(\mathcal{H}_{\bar{u}} \setminus \{h_{\bar{u}}^t\}, \bar{u})$.

Yes/No 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 $\mathcal{W}_{\bar{u}}^t = \mathcal{W}(\mathcal{H}_{\bar{u}}, \bar{u}) \setminus \{[e_{\bar{u}}^t, l_{\bar{u}}^t]\}$ will enforce a decrease in the number of time-classes for this user.

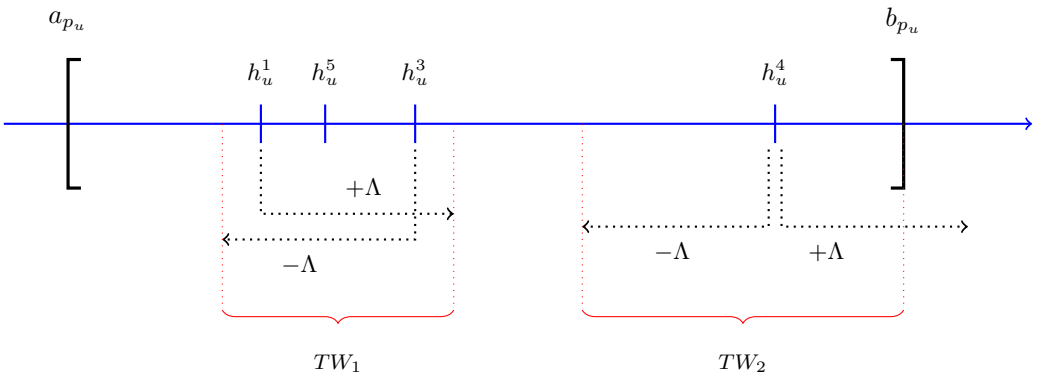


Figure 6: Construction of $\mathcal{W}(\mathcal{H}, u) = \{TW_1, TW_2\}$ for user u with pickup time windows $[a_{p_u}, b_{p_u}]$ and $\mathcal{H} = \{h_u^1, h_u^5, h_u^3, h_u^4\}$.

No/Yes In this case, the service time h_u^t defines a time-class on its own, so $\mathcal{W}_u^t = \{[a_{p_u}, b_{p_u}]\}$. As a result, any feasible service time at period t can be accepted.

No/No We distinguish two possibilities. In the first one, $\mathcal{W}_u^t$ is defined similarly as in case Yes/No. This possibility is selected with a probability $\nu = \theta \times (C_{\max} - \mu(\mathcal{H}_u))$, where $C_{\max}$ is the maximum number of time-classes in the current solution, and θ is a fixed parameter. In the second one (selected with probability $1 - \nu$), time windows $\mathcal{W}_u^t$ are set as in Case No/Yes. This may generate routes that increase the number of time-classes of this user. However, as in Feillet et al. [6], we observe that this temporary relaxation helps decrease the number of time-classes of user $\bar{u}$.

These four cases are summarized in Table 4.

		Q2: $\mu(\mathcal{H}_u \setminus \{h_u^t\}) < \mu(\mathcal{H}_u)$**?**	
		Yes	**No**
Q1: $u = \bar{u}$**?**	**Yes**	$\mathcal{W}_{\bar{u}}^t = \mathcal{W}(\mathcal{H}_{\bar{u}} \setminus \{h_{\bar{u}}^t\}, \bar{u})$	$\mathcal{W}_{\bar{u}}^t = \mathcal{W}(\mathcal{H}_{\bar{u}}, \bar{u}) \setminus \{[e_{\bar{u}}^t, l_{\bar{u}}^t]\}$
	No	$\mathcal{W}_u^t = \{[a_{p_u}, b_{p_u}]\}$	With probability ν: $\mathcal{W}_u^t = \mathcal{W}(\mathcal{H}_u, u) \setminus \{[e_u^t, l_u^t]\}$ Otherwise: $\mathcal{W}_u^t = \{[a_{p_u}, b_{p_u}]\}$

Table 4: Overview of cases for the definition of $\mathcal{W}_u^t$.

5.3 Solving the DARPmTW

This section details Line 10 of Algorithm 3. It consists in solving a DARP with multiple time windows (DARPmTW) as defined in the previous section. To the best of our knowledge the DARPmTW has not been treated in the literature.

We adapt the LNS-SCP used in Section 4.2 to solve single period DARPs with one time window at each pickup or delivery point. As 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.

In this LNS, each route duration is minimized and the DARP temporal constraints are checked through a route scheduling algorithm (called *the DARP scheduling algorithm*). More specifically, let us consider an insertion check in a route $\omega \in \Omega$. The DARP scheduling algorithm considers a maximum ride time $\bar{T}_u$, a pickup time window $[a_{p_u}, b_{p_u}]$ and a delivery time window $[a_{d_u}, b_{d_u}]$ for each user u visited by the route. It checks temporal constraints and if the route is feasible, the algorithm computes a service time h_i for each vertex such that: (i) the route duration is minimized and (ii) service times are scheduled as early as possible within the vertex time windows. In this algorithm, a maximum route time shift Δ_ω^+ is computed. This value indicates how much the route schedule can be shifted forward, while preserving

both its feasibility and the route duration. Note that since the duration of the route is minimized, the route includes waiting times only when Δ_ω^+ is zero. Otherwise it would be possible to reduce the route duration by postponing its departure time. Hence, when Δ_ω^+ is strictly positive, there is no waiting time on the route and shifting its departure by a value δ shifts the service times of all vertices on the route by the same value. The detailed procedure, proposed in Tellez et al. [29], 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, only deliveries will be checked.

Algorithm 4: Scheduling algorithm for a route ω for which additional multiple time windows have been defined

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 not 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 not scheduledUser then
15         return unfeasible
16 return feasible
```

Let $\mathcal{U}_\omega$ be the set of users ranked according to the 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.

First, the route time shift δ is initialized to 0 (line 1). 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 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 $\mathcal{O}(|\mathcal{U}_\omega| \sum_{u \in \mathcal{U}_\omega} |\mathcal{W}_u^t|)$.

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. Section 6.3 evaluates the main components of the matheuristic approach. In Section 6.4, SPεC is assessed on benchmark instances of [8] and [6] for the time-consistent VRP. Finally, Section 6.5 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 5 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
$N = 100 \times \lceil \lvert\mathcal{U}\rvert/100 \rceil$	number of routes to be appended to the pool at each call of the TC-DARP
$N^{init} = N \times \lvert\mathcal{T}\rvert$	number of routes to be appended to the pool at each call of the MP-DARP
$t_{\max} = 60s$	MILP solver time limit
$N_{\max} = 50000$	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 selection rule
$\rho = 6$	Random Biased Selection parameter

Table 5: SPεC parameters (all defined in Sections 3-5).

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. In the same way, parameter N^{init} was determined to solve the MP-DARP. Note that parameter N^{init} is much bigger than N because the MP-DARP contains less binary variables (so is an easier problem to solve) than the TC-DARP.

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 50000 routes and to 5000 routes, respectively.

The choice of selection rules for the MP-DARP and route policies for the TC-DARP are the following:

- Selection rules MP-DARP: $\mathcal{R} = \{SS\}$
- Route policies TC-DARP: $\mathcal{O} = (\{\mathcal{L}, SS\}, \{\mathcal{L}_{new}, SS\})$

SS stands for the sequential selection rule and $\mathcal{L}_{new}$ the pool generated through the DARP with the multiple time windows (see Section 4.2).

Some of the experiments that lead to these parameter choices are presented in Section 6.3.

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 people (users) to MSIs. We decomposed this data according to geographical areas in three different ways and built 8 small instances with 60 to 80 users, 4 medium-size instances with 120 to 160 users, and 2 large instances with 280 to 295 users. We assume an infinite homogeneous fleet; each vehicle has a capacity of 4 seats and 3 wheelchair spaces. Vehicle costs are composed of an hourly cost α =23.8€ and a cost per kilometer τ =0.17€. Additionally, we artificially set an arbitrarily small vehicle fixed cost of $\lambda = 1$ in order to favor solutions with the same variable cost but fewer vehicles.

Travel times and distances are obtained from the Open Source Routing Machine[2] (OSRM) proposed by Luxen and Vetter [17]. 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 Evaluation of the matheuristic components

In this section we present the tests that were used to evaluate the main components of the SPεC matheuristic. We compare several settings of the algorithm on a representative sample of five representative instances. Five runs are performed for each instance and setting.

Three variants are tested, each of which allows us to evaluate key components of the algorithm:

- The SPεC variant is the default configuration including all parameters defined as presented in Section 6.1.
- The SPεC-noDARPmTW variant corresponds to SPεC without the new routes generation procedure presented in Section 5.3, e.g. $\mathcal{O} = \{(\mathcal{L}, SS)\}$.
- The SPεC-withRBS variant corresponds to SPεC, plus the Random Biased Selection rule presented in Section 4.4.2, e.g. $\mathcal{O} = \{(\mathcal{L}, SS); (\mathcal{L}, RBS); (\mathcal{L}_{new}, SS)\}$.

6.3.1 Comparison based on cost per time-classes

We first compare each variant based on the best solution cost it is able to find when the number of time-classes is limited to 3, 2 or 1 for each user. Note that although our instances have 5 time periods, users in non-dominated solutions present at most 3 time-classes. Table 6 reports the corresponding computational results.

Instance names are reported in Column 1. Columns 2-4 present the value of the objective function of the best solutions found, regardless the run and the setting. The best results are shown in bold. In the next columns we report the relative gap between the average cost obtained with the corresponding selection rule and the best solution found (Best). It is computed as (Avg Cost - Best) / Best $\times$ 100. Row Avg reports the average value of the corresponding column.

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

Instance	Best			SPεC-noDARPmTW			SPεC-withRBS			SPεC		
$C_{\max}$	3	2	1	3	2	1	3	2	1	3	2	1
C01_60	1291.03	1291.73	1345.41	**0.11%**	2.15%	19.20%	**0.11%**	**0.58%**	5.17%	**0.11%**	**0.32%**	**2.52%**
C02_80	2523.21	2528.83	2561.40	**0.01%**	**0.20%**	**3.74%**	**0.01%**	0.26%	1.09%	**0.01%**	0.40%	**0.49%**
C09_135	2857.18	2863.86	3107.12	**0.34%**	**0.93%**	15.46%	**0.34%**	0.99%	8.69%	**0.34%**	1.22%	**3.74%**
C10_160	2621.34	2621.34	2779.06	0.19%	1.04%	19.75%	**0.14%**	0.78%	10.07%	0.15%	**0.70%**	**5.88%**
C12_280	7712.22	7730.04	9487.86	**0.26%**	**0.54%**	10.25%	**0.26%**	0.82%	**3.55%**	0.27%	1.22%	3.67%
			Avg	0.18%	0.97%	13.68%	**0.17%**	**0.69%**	5.72%	**0.17%**	0.77%	**3.26%**

Table 6: Average gaps of SPεC variants for solutions with a $C_{\max}$ of 3, 2 and 1 time-classes.

The first observation of Table 6 is that average values (Avg) among variants differ significantly only in solutions with one time-class $C_{\max} = 1$. Comparing SPεC-noDARPmTW to SPεC, we find that the DARPmTW component brings significant improvement in the quality of solutions with one time-class. New routes have to be generated to improve consistency. Comparing the SPεC-withRBS to SPεC, we seen that the Random Biased Selection procedure does not bring significant improvement in solutions with three and two time-classes, and worsens the performance of one time-class solutions. According to these experiments, we conclude that the SPεC configuration seems to outperform the other variants when looking at the best cost found for each number of time-classes.

6.3.2 Comparison based on hypervolume indicators

To complete the previous experiments, we provide deeper insights on the quality of the Pareto front approximations that are produced by each configuration. Since we try to find a good balance between costs and quality of service, we are particularly interested in solutions with at most two time-classes per passenger. We compare the hypervolume indicators of the Pareto front approximations found by each configuration.

The hypervolume indicator was introduced by Zitzler et al. [33] to compare Pareto front approximations in multi-objective optimization problems. It measures the volume between the set of non-dominated solutions and a reference point. In this study we use the Nadir point approximation as a reference point. For a given instance this point is an artificial point in the objective space. It takes as coordinates the cost of the best solution that has one time-class for each user and the consistency level of the minimum cost non-dominated solution. The larger the hypervolume indicator, the better the approximation [33].

Table 7 reports the hypervolume indicators for the instances and configurations that were presented in the previous section.

		Avg Gap hypervolume		
Instance	Best	SPεC-noDARPmTW	SPεC-withRBS	SPεC
C01_60	9979.72	28.93%	3.91%	**1.88%**
C02_80	2788.82	19.04%	5.61%	**4.06%**
C09_135	32847.86	20.34%	5.94%	**5.36%**
C10_160	32969.07	26.48%	4.68%	**3.25%**
C12_280	253897.10	13.42%	**6.49%**	6.82%
Avg		21.64%	5.33%	4.28%

Table 7: Avg Gap(%) with respect to the best hypervolume (5 runs)

The instance name is reported in Column 1. Column 2 presents the best hypervolume found regardless of the run and the setting. The best results are shown in bold. In the next columns, we report the relative gap between the average value hypervolumes with the corresponding setting and the best hypervolume found (Best). It is computed as ((Avg hyperpervolume - Best) / Best) $\times$ 100. Row Avg is the average value of the corresponding column.

The first observation of Table 7 is that SPεC has the best results in 4 out of 5 instances. The SPεC-withRBS provides slightly better results for only one instance. Comparing SPεC-noDARPmTW against SPεC, we once more conclude that the DARPmTW component is needed to generate routes that significantly improve the solution quality.

Overall, we can confirm with this experiment that the SPεC configuration outperforms other variants. To illustrate this finding, Figure 7 shows three Pareto front approximations obtained with the

SPεC configurations for instance I01_60.

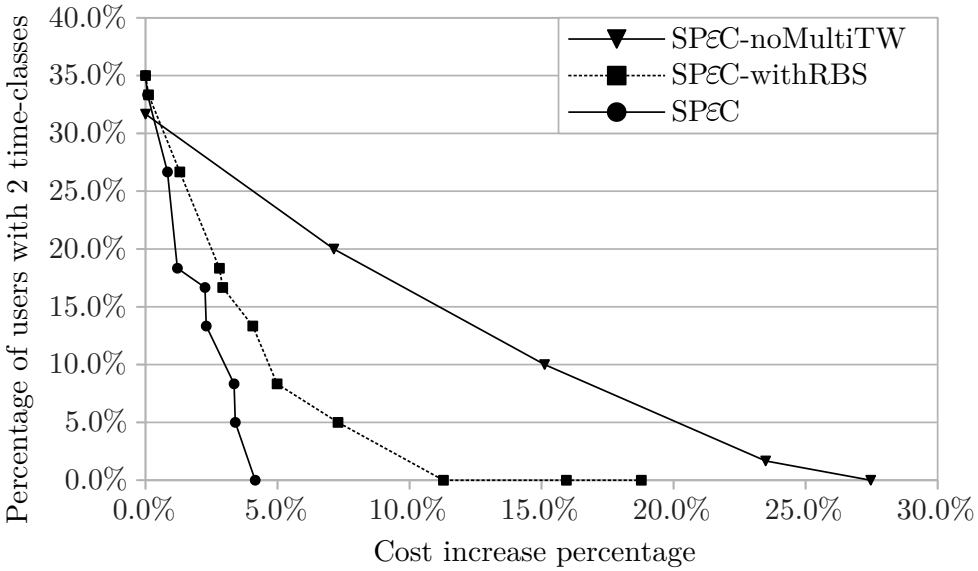


Figure 7: Comparison of Pareto fronts for instance C01_60.

6.4 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. [8]. This adaptation, proposed by Feillet et al. [6], transforms the small conVRP instances of Groër et al. [8] 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. [6]. 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 be traveled in either direction. Thus, each time a route is appended to the pool, the reverse route is also appended.

6.4.1 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 8 presents the comparison with SPεC over 10 runs.

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%.

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 8: Benchmark on RconVRP instances reported in [6]

6.4.2 Instances TC-VRP

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. [6] 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 9 and 10 show the average gap of SPεC with respect to the LNS method of Feillet et al. [6], 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 9 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%.

Instance	Avg Gap Transportation cost				
$C_{\max}$	$\leq$5	$\leq$4	$\leq$3	$\leq$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 9: Results aggregated by percentage of user requests on the benchmark of Feillet et al. [6]

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 BKW) which is more than 50 for solutions above 3 time-classes, and 35 for solutions with 1 and 2 time-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 10 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				
$C_{\max}$	$\leq$5	$\leq$4	$\leq$3	$\leq$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 10: Results aggregated by instance size on [6] benchmark

6.5 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 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 size, the increase of cost for reaching single time-class solutions can vary from 4% in C01_60 to 23% in the C12_280 instance. 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.

6.6 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. [11] 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

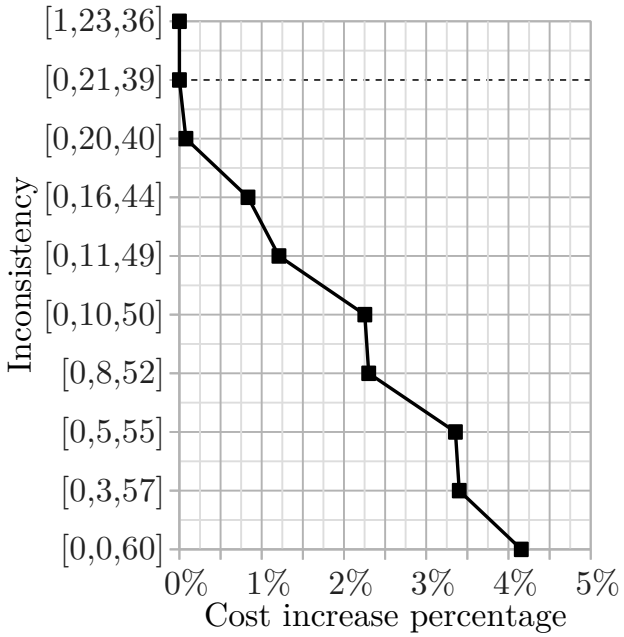


Figure 8: Pareto front approximation for instance C01_60

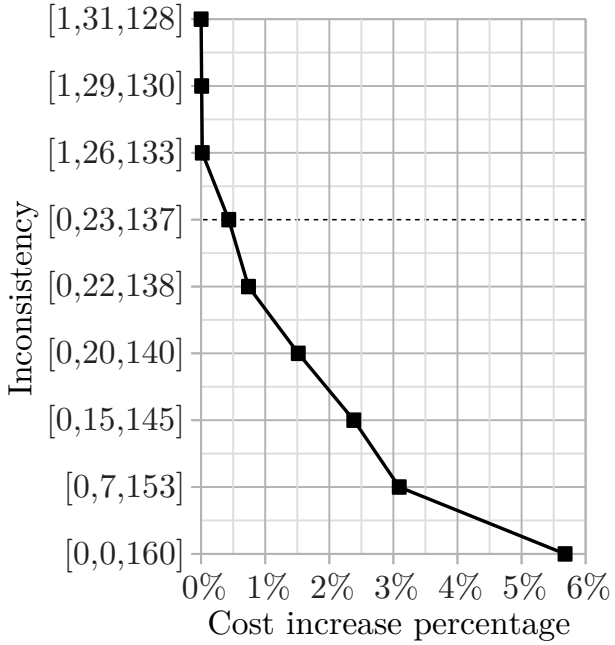


Figure 9: Pareto front approximation for instance C10_160

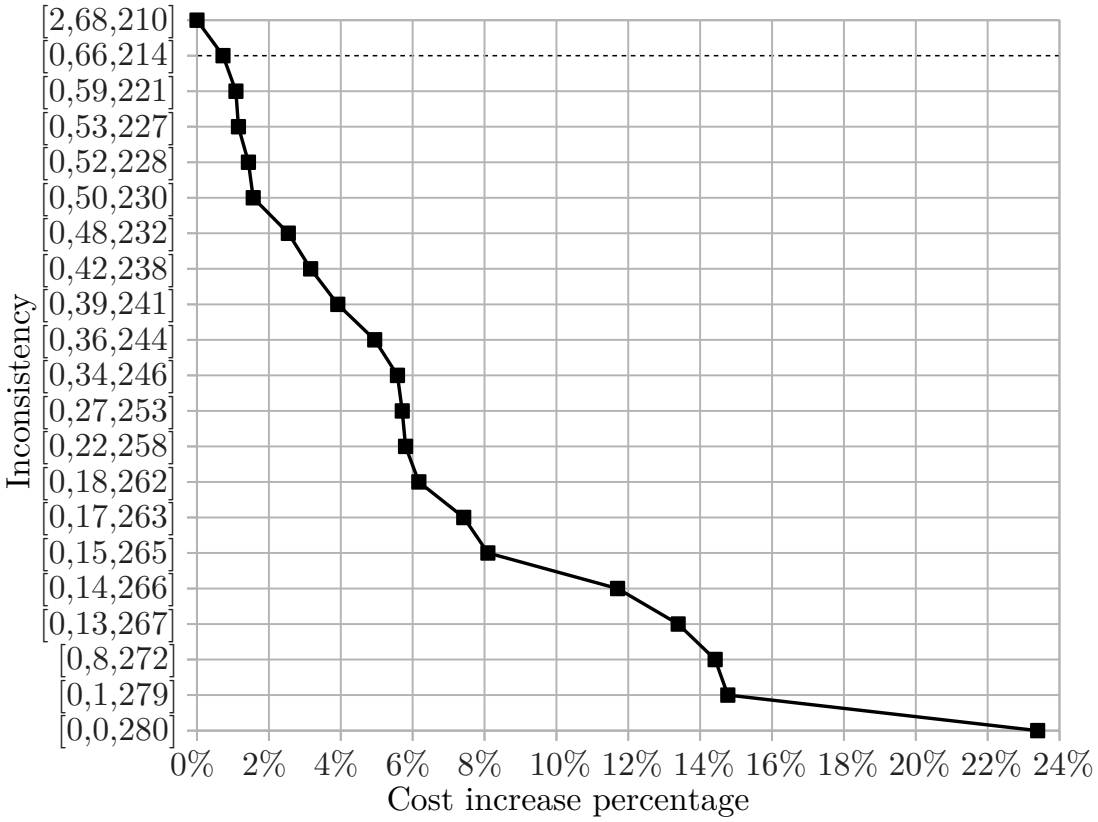


Figure 10: Pareto front approximation for instance C12_280

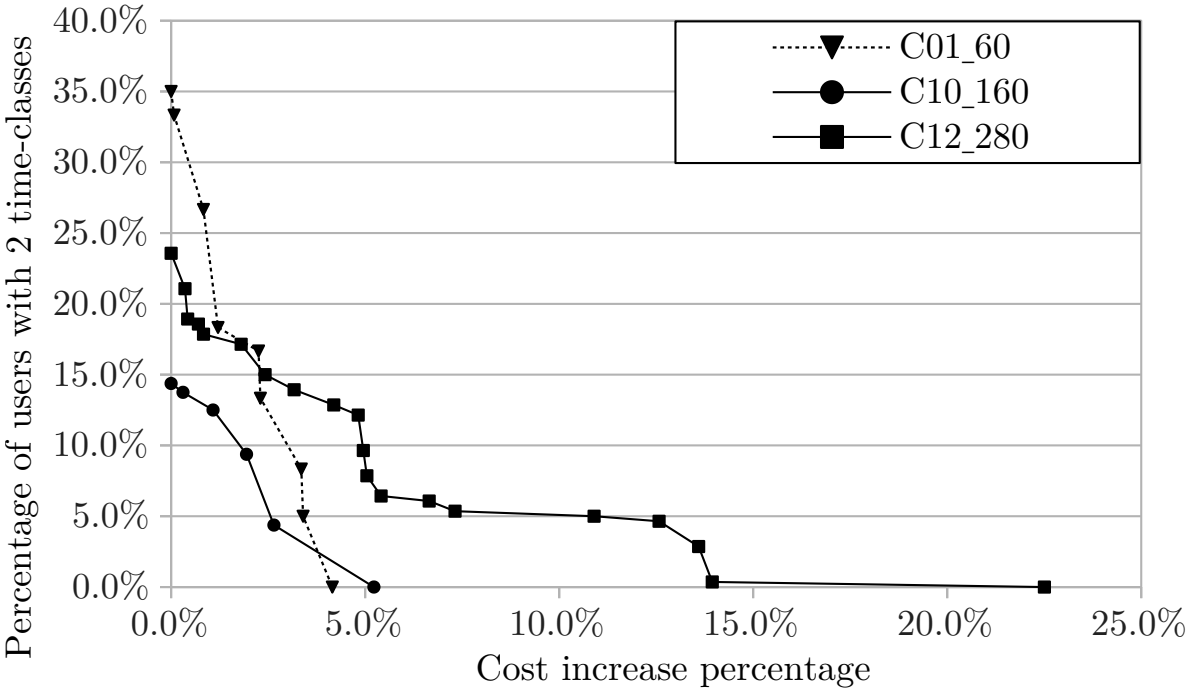


Figure 11: Scaled representation of the Pareto front approximations

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.

Instance	SPεC				SPεC($\Delta_\omega^+ = 0$)		
$C_{\max}$	3	2	1	%R-Shift	3	2	1
C00_80	2045.54	2044.43	2061.89	49%	-0.07%	0.25%	0.35%
C01_60	1291.03	1291.73	1345.41	40%	0.18%	0.48%	0.46%
C02_80	2523.21	2533.86	2561.40	31%	0.00%	-0.01%	0.31%
C03_70		1735.07	1752.43	33%		0.00%	0.30%
C04_80		1207.72	1220.91	32%		0.00%	2.44%
C05_80	1871.25	1871.53	1923.21	40%	0.37%	0.06%	0.08%
C06_60		3304.41	3332.21	42%		0.00%	0.54%
C07_65	1865.46	1868.15	1920.95	50%	0.00%	-0.07%	3.22%
C08_120		5500.45	5534.44	39%		0.02%	1.76%
C09_135	2857.18	2880.57	3107.12	38%	0.27%	-0.18%	1.17%
C10_160	2621.35	2621.35	2779.06	42%	0.03%	0.00%	8.68%
C11_160	3222.10	3233.17	3549.53	42%	0.21%	0.00%	4.84%
C12_280	7712.44	7740.33	9487.86	33%	0.02%	-0.22%	4.64%
C13_295	6294.89	6404.49	7664.17	47%	-0.04%	-1.05%	6.10%
			Avg	40%	**0.10%**	**-0.05%**	**2.49%**

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

The average gain is found only for one time-class solutions with a 2.49% saving.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.

7 Conclusions

This paper introduces a new variant of the multi-period 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. The transportation cost includes routing and vehicle ownership costs. The time-consistency objective is expressed as a lexicographic function of the number of users having significantly different service times. Regarding the literature on the topic, this is a new formulation of the time-consistency of the solution, which generalizes the approach of Feillet et al. [6]. 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 Auvergne-Rhône-Alpes, a region of France.

To compute Pareto front approximations, we developed a matheuristic framework called SPεCbased 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. [29]. 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. Different selection rules are presented to this end. 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 passengers with disabilities are very sensitive to inconsistent schedules. In this study, we show that economic solutions are already fairly consistent with very few passengers having 3 time-classes. In addition, we found that in most instances, with a small increase in transportation costs ($<1\%$), passenger 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 will be model extensions to add driver-related constraints such as working time, breaks, regulations and driver-consistency. As regards vehicles, our model can be extended to a 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 Construction of $\mathcal{W}(\mathcal{H}, u)$

Algorithm 5: Construction of $\mathcal{W}(\mathcal{H}, u)$

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 */
```
$\overline{h} = h_1$
$\underline{h} = h_1$
for $i = 2, \ldots, M$ **do**
  if $h_i > \underline{h} + \Lambda$ **then**
    mtw $\leftarrow$ mtw $\cup \{[\max\{a_{p_u}; \overline{h} - \Lambda\}, \min\{b_{p_u}; \underline{h} + \Lambda\}]\}$
    $\underline{h} = h_i$
  $\overline{h} = h_i$
mtw $\leftarrow$ mtw $\cup \{[\max\{a_{p_u}; \overline{h} - \Lambda\}, \min\{b_{p_u}; \underline{h} + \Lambda\}]\}$
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

  $H \leftarrow H + \max\{0; a_i - (h_{i-1} + t_{i-1,i} + \zeta_{i-1})\}$

  $F' = F$

  $F \leftarrow \min\{F; H + \max\{0; l_i - h_i\}\}$

  if $i = M$ **then**

    $F' \leftarrow \min\{F'; H\}$

/* Phase 2: optimize route duration */

$\Delta_\omega^+ = F - F'$ /* 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 $i \in \mathcal{P}$ **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 $\{h_i | i \in \omega\}, \Delta_\omega^+$

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 [29].

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 [6]

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 [6]

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

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

[5] 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.

[6] 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.

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

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

[9] 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.

[10] 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.

[11] 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.

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

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

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

[20] 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.

[21] 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.

[22] 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.

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

[24] 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.

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

[26] 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.

[27] 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.

[28] 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.

[29] 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.

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

[31] 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.

[32] 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.

[33] Eckart Zitzler et al. "Performance assessment of multiobjective optimizers: An analysis and review". In: *TIK-Report* 139 (2002).