CIRRELT

CIRRELT-2026-15

Polynomial Time Separation Algorithms for Robust Vehicle Routing under Travel-time Uncertainty

Rafael Ajudarte de Campos
Leandro C. Coelho
Pedro Munari

August 2026

Bureau de Montréal
Université de Montréal
C.P. 6128, succ. Centre-Ville
Montréal (Québec) H3C 3J7
Tél : 1-514-343-7575
Télécopie : 1-514-343-7121

Bureau de Québec
Université Laval,
2325, rue de la Terrasse
Pavillon Palasis-Prince, local 2415
Québec (Québec) G1V 0A6
Tél : 1-418-656-2073
Télécopie : 1-418-656-2624

Polynomial Time Separation Algorithms for Robust Vehicle Routing under Travel-time Uncertainty

Rafael Ajudarte de Campos[1*], Leandro C. Coelho[1-2], Pedro Munari[3]

[1] Faculté des Sciences de l'Administration, Université Laval, Québec, Canada

[2] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT).

[3] Department of Production Engineering, Federal University of São Carlos (UFSCar), São Carlos, São Paulo, Brazil

Abstract: We study the vehicle routing problem with time windows (VRPTW) under travel-time uncertainty considering different uncertainty sets. We propose a new polynomial time separation algorithm that can be used with any compact convex uncertainty set and easily incorporated into branch-and-cut (BC), branch-price-and-cut (BPC), or heuristic algorithms. This flexibility distinguishes the proposed algorithm from previous works in the robust VRPTW, which mostly rely on the traditional cardinality-constrained uncertainty set or, more recently, on the knapsack uncertainty set. In addition to these uncertainty sets, we consider the ellipsoidal, factor model, and discrete uncertainty sets using instances from the VRPTW literature. We develop tailored BC and BPC methods to validate the proposed separation algorithm for these sets. Finally, we evaluate the impact of the studied uncertainty sets on the trade-off between the robustness of the solution and its cost.

Keywords: vehicle routing problem, time windows, uncertainty, robust optimization, time uncertainty.

This work was partly supported by the Canadian Natural Sciences and Engineering Research Council (NSERC) [grant number 2019-00094], the Fonds de recherche du Québec - Nature et technologie [programme Bourses de doctorat en recherche, dossier 347556], the São Paulo Research Foundation (FAPESP) [grant numbers 13/07375-0, 19/22235-6, 22/05803-3], the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) [Finance Code 001], and the National Council for Scientific and Technological Development (CNPq) [grant number 314079/2023-8]. We thank the Digital Research Alliance of Canada for providing high-performance parallel computing resources.

---

* Corresponding author: rafael.ajudarte-de-campos.1@ulaval.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2026

1 Introduction

The use of deterministic approaches to solve the vehicle routing problem with time windows (VRPTW) often leads to routes that are sensitive to variations in travel times, requiring significant adjustments at execution time once the uncertain parameters become known [1, 2, 3]. These adjustments create operational challenges, as they may require last minute decisions, increase management complexity, and incur additional, unanticipated costs. Travel times can be affected by many factors, such as the weather and traffic, resulting in delays that might make the vehicle fail to serve a customer within the planned time windows. Nevertheless, most vehicle routing literature considers that all parameters are known a priori, even though optimal solutions based on these values are likely to become infeasible in practice even for small variations in travel times [4, 5, 6].

Several methods aim to incorporate these uncertainties into models and solution algorithms. In this paper, we focus on Robust Optimization (RO), which has gained increased attention in the VRP literature in the last decade. Compared to stochastic programming approaches, RO has the advantage of not requiring knowledge of the probability distribution of the uncertain parameters. Instead, it relies on bounds on their possible realizations, such as the maximum travel time between two nodes [4]. In particular, the use of RO to incorporate travel-time uncertainty in a vehicle routing problem (VRP) is relatively new. Lee et al. [7] proposed an exact algorithm based on column generation (CG) to solve the robust VRP with deadlines under demand and travel-time uncertainties, solving instances with up to 25 customers. The robust VRP with deadlines is a variant of the robust VRPTW (RVRPTW) in which time windows are defined by their closing time only (i.e., they open at the beginning of the time horizon). Despite this similarity between these two problems, the authors mention that it is not straightforward to extend their algorithm to solve the RVRPTW.

Agra et al. [8] proposed the first RO formulation for the RVRPTW under travel-time uncertainty. They proposed a model for a problem with a heterogeneous fleet, but no capacity constraints, solving instances with up to 20 customers. This problem was later revisited by Agra et al. [9], who proposed two more efficient robust approaches, capable of solving instances with up to 50 customers, one extending the resource inequalities formulation by employing adjustable RO and the other generalizing a path inequality formulation to consider uncertainty on travel times. The first formulation proposed by Agra et al. [9] was later adapted by Hu et al. [10], who also proposed a two-stage heuristic based on a modified adaptive variable neighborhood search to solve the RVRPTW with uncertainty on demand and travel times. This heuristic solved instances with up to 100 customers in a reasonable time.

Munari et al. [6] proposed a compact model for the RVRPTW under uncertainty on demand and travel times based on the linearization of recursive equations. This approach showed superior computational results to dualization-based compact formulations [8, 7]. Additionally, the authors proposed a branch-price-and-cut (BPC) method based on a set partitioning formulation, solving most of the benchmark instances with up to 100 customers. Wang et al. [11] also used a BPC algorithm to solve the robust capacitated vehicle routing problem (RCVRP) and the RVRPTW. The authors proposed a cutting-plane framework implemented within a BPC solver [12], which allows the temporary generation of non-robust feasible columns that are subsequently eliminated using infeasible path elimination constraints. Abreu et al. [13] introduced the robust pickup and delivery problem with time windows under demand and travel-time uncertainty, proposing compact and cutting-plane formulations as well as tailored branch-and-cut (BC) and BPC algorithms based on them.

Recently, Bartolini et al. [14] developed an exact algorithm based on CG and dynamic programming for the robust traveling salesman problem with time windows with uncertain travel times, which solved instances with up to 80 customers. In addition to the cardinality-constrained set, they also use the single-knapsack uncertainty set to model travel-time uncertainty. To the best of our knowledge, they were the first to model travel-time uncertainty in a problem with time windows (with nonzero opening times) using a set other than the cardinality-constrained set [15]. The knapsack uncertainty set was also explored by Campos et al. [16], who proposed new compact formulations based on commodity flow constraints and a BC algorithm to solve the RVRPTW under demand and travel-time uncertainty. To incorporate uncertainty, the authors proposed

a formulation based on the linearization of recursive equations.

While demand uncertainty is common in routing problems under different uncertainty sets, such as the ellipsoidal [17, 18], the cardinality-constrained [6, 18, 19], the knapsack [5, 20, 18, 21], the factor model [5, 18], and the discrete set [17, 18], studies considering uncertainty on travel times are usually limited to the cardinality-constrained and the discrete uncertainty sets. As already mentioned, only recently the *single* knapsack uncertainty set was used to represent travel times [14, 16]. Thus, there is a noticeable gap in the literature as no current solution method can consider uncertainty on travel times for the other sets.

The lack of research addressing uncertainty on travel times is a consequence of the additional difficulty introduced by the earliest service start time at the customers. More specifically, to evaluate whether a solution for problems with deviation on the demand is robust feasible, one only needs to find a combination of worst-case realizations within the uncertainty set that results in the largest absolute deviation, regardless of their sequence. This does not hold for problems with time windows and uncertainty on travel times. The uncertainty realization that leads to the maximum combination of delays does not necessarily lead to the latest arrival time, as deviation in some arcs can be absorbed by the waiting time required before the time window opens.

The consideration of alternative uncertainty sets is appealing for two primary reasons. First, different sets may be more effective at capturing the system characteristics and more readily implementable. Second, their structure can yield solutions with distinct trade-offs between solution cost and robustness. In particular, certain uncertainty set configurations can yield solutions that are not found by other sets. Therefore, the ability to model and incorporate different uncertainty sets without a significant increase in computational effort could greatly expand the range of robust solutions available to decision-makers.

Building on this motivation, the main contributions of this paper are summarized as follows:

- A polynomial time separation algorithm for the RVRPTW under travel-time uncertainty, capable of verifying the robust feasibility of a given solution for any compact convex uncertainty set;
- Closed-form expressions for five commonly used uncertainty sets in the literature, cardinality-constrained, axis-parallel ellipsoidal, knapsack, factor model, and discrete sets, showing how the proposed algorithm can be efficiently applied to each of them;
- Exact BC and BPC algorithms that dynamically integrate the proposed separation procedure to ensure robust feasibility;
- Computational enhancements that exploit the structure of the robust problem, including the first robust extensions of reachability cuts for the RVRPTW, time-window tightening and arc-removal procedures for each studied uncertainty set;
- Extensive computational study on instances adapted from the literature, analyzing the impact of each uncertainty set on solution cost, robustness, and computational performance.

The remainder of this paper is organized as follows. Section 2 formally introduces the RVRPTW and the uncertainty sets considered in this study. Section 3 describes the proposed separation algorithm to efficiently characterize the feasibility of a route under travel-time uncertainty. Section 4 presents the BC framework developed for the RVRPTW, while Section 5 introduces the BPC algorithm, which also integrates the proposed separation algorithm. Section 6 reports the results of extensive computational experiments, highlighting the impact of each uncertainty set on the solution. Finally, Section 7 provides concluding remarks and outlines future research directions.

2 Problem definition and uncertainty sets

To represent the deterministic and robust versions of the VRPTW, we define a complete directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = \{0, 1, \ldots, n+1\}$ is the set of nodes connected by a set of arcs $\mathcal{A} = \{(i, j) : i, j \in$

$\mathcal{N}, i \neq j, i < n+1, j > 0\}$. The subset $\mathcal{N}^* = \{1, \ldots, n\}$ corresponds to the customers, and nodes 0 and $n+1$ represent the departure and arrival depots, respectively. This distinction is a modeling technique, and a physical separation between the depots is not required. The depot houses a homogeneous fleet with V vehicles of capacity Q. Each node $i \in \mathcal{N}$ is associated with a demand d_i, a service time s_i, and a time window $[w_i^a, w_i^b]$. This time window represents the time a node can be serviced by a vehicle; if the vehicle arrives before w_i^a, it must wait for the time window to open to start the service, and it is not allowed to start servicing after w_i^b. We assume that $d_0 = d_{n+1} = 0$, $s_0 = s_{n+1} = 0$, $w_0^a = w_{n+1}^a = 0$, and $w_0^b = w_{n+1}^b$ is provided by the decision-maker.

2.1 Deterministic problem

In the deterministic VRPTW, each arc $(i, j) \in \mathcal{A}$ is associated with a known travel time t_{ij} and a travel cost c_{ij}. The objective is to determine the least-cost routes to service all customers exactly once while respecting capacity and time window constraints. We use three sets of decision variables to model this problem: the binary x_{ij}, which represents if arc $(i, j) \in \mathcal{A}$ is traversed in the solution, and continuous variables u_i and w_i which represent, respectively, the load in the vehicle departing from node $i \in \mathcal{N}$ and the service starting time in that node. Using the defined parameters and decision variables, a model to represent the VRPTW is as follows:

$$\min \sum_{(i,j)\in\mathcal{A}} c_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \sum_{(i,j)\in\mathcal{A}} x_{ij} = 1, \qquad i \in \mathcal{N}^*, \tag{2}$$

$$\sum_{(i,h)\in\mathcal{A}} x_{ih} = \sum_{(h,j)\in\mathcal{A}} x_{hj}, \qquad h \in \mathcal{N}^*, \tag{3}$$

$$\sum_{(0,j)\in\mathcal{A}} x_{0j} \leq V, \tag{4}$$

$$u_j \geq u_i + d_j - Q(1 - x_{ij}), \qquad (i, j) \in \mathcal{A}, \tag{5}$$

$$u_0 = 0, \tag{6}$$

$$u_j \leq Q, \qquad j \in \mathcal{N}, \tag{7}$$

$$w_j \geq w_i + t_{ij} + s_i - M_{ij}(1 - x_{ij}), \qquad (i, j) \in \mathcal{A}, \tag{8}$$

$$w_i^a \leq w_i \leq w_i^b, \qquad i \in \mathcal{N}, \tag{9}$$

$$u_j, w_j \in \mathbb{R}^+, \qquad j \in \mathcal{N}, \tag{10}$$

$$x_{ij} \in \{0, 1\}, \qquad (i, j) \in \mathcal{A}. \tag{11}$$

The objective function (1) seeks to minimize the total transportation cost. Constraints (2) ensure that each node i is visited exactly once. Constraints (3) impose that a vehicle can depart from node i only if a vehicle arrives at this node. Constraint (4) guarantees that at most V vehicles leave the depot. Constraints (5) propagate the load carried on the vehicle arriving at node j and forbid subtours. Constraint (6) imposes the initial vehicle load. The capacity of the vehicles is imposed by constraints (7). Constraints (8) propagate the arrival time at node j, where M_{ij} is a sufficiently large value defined as $M_{ij} = \max\{0, w_i^b + t_{ij} + s_i - w_j^a\}$. Time windows are enforced by constraints (9). Finally, constraints (10)–(11) define the domain of the decision variables.

2.2 Uncertainty sets

In the robust counterparts of the VRPTW, the travel times t_{ij} are no longer precisely known a priori. Instead, these parameters are treated as random variables whose values range around a nominal travel time $\bar{t}_{ij}$ to a maximum absolute deviation $\hat{t}_{ij}$, i.e., $t_{ij} \in [\bar{t}_{ij} - \hat{t}_{ij}, \bar{t}_{ij} + \hat{t}_{ij}]$. To avoid over-conservative solutions, uncertainty sets model the total deviation that can simultaneously happen in a solution, thus controlling its robustness. We now describe the five uncertainty sets used in this work.

2.2.1 Cardinality-constrained uncertainty set (U_Γ).

This is the most widely used uncertainty set in the robust optimization literature. Introduced by Bertsimas and Sim [15], this uncertainty set quickly became the staple in RO literature for being the only controllable one that could be implemented via linear constraints at the time. Ordonez [4] was the first to model a robust VRP under demand uncertainty using this uncertainty set. Later, Lee et al. [7] use the cardinality-constrained set to model travel-time uncertainty in the robust VRP with deadlines.

In this set, the decision-maker controls the robustness of the solution by choosing a budget Γ that limits the threshold of the total scaled variation of the uncertain parameters. Intuitively, Γ can be interpreted as the number of components of the uncertain parameter that attain their worst-case value. It is based on the premise that not all uncertain components (e.g., travel-time deviation on each arc), will attain their worst-case value simultaneously. For instance, choosing $\Gamma = 2$ implies that the solution must be feasible when the travel times of up to two arcs attain their worst-case value. When Γ is not an integer, the robust solution must be feasible when any $\lfloor \Gamma \rfloor$ arcs reach their worst-case deviations, and one additional arc deviates by a fraction $(\Gamma - \lfloor \Gamma \rfloor)$ of its maximum deviation $\hat{t}_{ij}$ [22]. Let the random variable $\xi^c_{ij} \in [0, 1]$ represent the fraction of the maximum absolute deviation of arc (i, j) considered for a given realization within the uncertainty set. Then, we can represent this set as follows:

$$U_\Gamma = \{t \in \mathbb{R}^{|\mathcal{A}|} \mid t_{ij} = \bar{t}_{ij} + \hat{t}_{ij}\xi^c_{ij}, (i,j) \in \mathcal{A}; \sum_{(i,j)\in\mathcal{A}} \xi^c_{ij} \leq \Gamma; \xi^c_{ij} \in [0,1], (i,j) \in \mathcal{A}\}.$$

2.2.2 Knapsack uncertainty set (U_Δ).

Instead of imposing a limit on the number of worst-case realizations, this set constrains the total deviation of the uncertain parameters [5]. Hence, to control the robustness of the solution using this uncertainty set, the decision-maker chooses a budget Δ representing the maximum absolute deviation allowed on a route. This parameter is often easier to estimate in practice than the budget Γ from the cardinality-constrained uncertainty set, as drivers typically have a better sense of how late they are than of how many individual trips between customers exceed expected durations.

Additionally, it is possible to partition the arcs into disjoint subsets (knapsacks) S_l, and assign an individual budget Δ_l to each knapsack $l \in L$, resulting in the multiple knapsack uncertainty set. This structure can be used to model, e.g., different geographical areas, which can be subject to variation independently. We then define $\xi^k_{ij} \in [0, 1]$, to represent the fraction of the maximum deviation on arc (i, j) for a given realization, and the subset S_l of arcs in each knapsack $l \in L$. With this information, this set can be represented by:

$$U_\Delta = \{t \in \mathbb{R}^{|\mathcal{A}|} \mid t_{ij} = \bar{t}_{ij} + \hat{t}_{ij}\xi^k_{ij}, (i,j) \in \mathcal{A}; \sum_{(i,j)\in S_l} \xi^k_{ij}\hat{t}_{ij} \leq \Delta_l, l \in L; \xi^k_{ij} \in [0,1], (i,j) \in \mathcal{A}\}.$$

2.2.3 Ellipsoidal uncertainty set (U_Ω).

Introduced by Ben-Tal and Nemirovski [23], it was the first set in the RO literature to allow the decision-maker to control the robustness of the solution. However, the nonlinear nature of this set has limited its application in the VRP literature [17, 18]. This set stipulates that the realization of the uncertain parameter (e.g., travel

time) should lie within an ellipsoid centered at the nominal travel-time vector $\bar{t}$, allowing deviations along multiple correlated directions, resulting in different axis inclinations.

A notable case for this set is the axis-parallel ellipsoidal set, where the ellipsoid's axes are aligned with the coordinate axes. In this configuration, the decision-maker may select a maximum radius $\Omega \leq 1$ from the center of the ellipsoid to its surface, with larger values yielding more conservative solutions. Let $\xi^e_{ij} \in [-1, 1]$ be a random variable that governs the deviation on arc (i, j) within the ellipsoid. The uncertainty set can then be defined as follows:

$$U_\Omega = \{t \in \mathbb{R}^{|\mathcal{A}|} \mid t_{ij} = \bar{t}_{ij} + \hat{t}_{ij}\xi^e_{ij}, (i,j) \in \mathcal{A};\ \sum_{(i,j)\in\mathcal{A}} (\xi^e_{ij})^2 \leq \Omega^2;\ \xi^e_{ij} \in [-1,1], (i,j) \in \mathcal{A}\}.$$

Note that, although ξ^e_{ij} may take negative values, the uncertainty set U_Ω is bounded by $\sum_{(i,j)\in\mathcal{A}}(\xi^e_{ij})^2 \leq \Omega^2$. Therefore, negative deviations would consume uncertainty budget without increasing travel times and can be safely disregarded.

2.2.4 Factor model uncertainty set (U_Ψ).

It was first introduced in the portfolio optimization literature [24] to incorporate estimation errors into the decision-making process. The uncertain parameter is represented as a linear combination of F independent factors within a factor loading matrix Ψ that correlates these factors to the uncertain parameter realization. For example, an entry $\Psi_{ijf} = 5$ of this matrix means that the total travel time on arc $(i, j) \in \mathcal{A}$ is increased by five units when factor f is fully present. Conversely, if the system exhibits behavior opposite to the expected condition associated with the factor (e.g., a cold day when the factor corresponds to hot weather), then t_{ij} may decrease by five units from its nominal value instead. The values of Ψ_{ijf} are scaled according to the maximum travel-time deviation on arc (i, j). In this work, we assume that $\sum_{f=1}^{F} |\Psi_{ijf}| \leq \hat{t}_{ij}, (i, j) \in \mathcal{A}$, so that the total deviation induced by the factors is bounded by the maximum deviation parameter.

The decision-maker must construct this matrix, typically using historical data and relying on statistical tools, such as principal component analysis or factor analysis. Let $\xi_f \in [-1, 1]$ be a random variable representing the magnitude and direction of factor f in a realization. The uncertainty budget is controlled through a parameter $\beta \in [0, 1]$, which imposes the constraint $-\beta F \leq \sum_{f=1}^{F} \xi_f \leq \beta F$. For example, setting $\beta = 0$ imposes that as many factors are positive as negative.

Then, the factor model can be represented by:

$$U_\Psi = \{t \in \mathbb{R}^{|\mathcal{A}|} \mid t_{ij} = \bar{t}_{ij} + \sum_{f=1}^{F} \xi_f \Psi_{ijf}, (i,j) \in \mathcal{A};\ -\beta F \leq \sum_{f=1}^{F} \xi_f \leq \beta F;\ \xi_f \in [-1,1], f = 1, \ldots, F\}.$$

Notice that, unlike the previous uncertainty sets, a realization $(\xi'_1, \ldots, \xi'_F)$ in which $t_{ij} < \bar{t}_{ij}$ for a given arc (i, j) may still lead to a worst-case scenario for the solution. This is because, although the travel time on arc (i, j) is reduced, the same realization may increase the travel times on other arcs (i.e., there may exist $(i', j') \in \mathcal{A}$ such that $\sum_{f=1}^{F} \xi'_f \Psi_{i'j'f} > 0$) which could ultimately increase the overall arrival time at a customer node.

2.2.5 Discrete uncertainty set (U_D).

In this set, the travel time takes values inside a convex hull of D scenarios specified by the decision-maker. Each scenario $s \in \{1, \ldots, D\}$ can be chosen considering the historical realization of the uncertain parameter. Although historical records are essential for creating a representative uncertainty set, one should be careful when selecting the data, as considering statistical outliers may result in over-conservative solutions. This uncertainty set can be represented by the following expression:

$$U_D = conv\{t^{(s)} \mid s = 1, \ldots, D\}.$$

3 Characterization of robust feasibility and separation algorithm

A solution is called robust feasible if it remains feasible for *all* realizations within the chosen uncertainty set. However, when working with the convex sets described in Section 2.2, evaluating all the infinite possible realizations is impractical. Instead, as we will demonstrate, it is sufficient to evaluate feasibility with respect to the extreme points of the uncertainty set. This is because any other realization would yield equal or lower total travel times compared to at least one of these extreme points. Hence, a solution is robust feasible for the VRPTW if, for every customer node, the worst-case arrival time, computed over the set of extreme points, respects the corresponding time window. Therefore, if one can efficiently determine the worst-case arrival times at all nodes of a route, one can also efficiently verify the solution's robust feasibility. This section presents an explicit characterization of the robust feasibility of a route for the RVRPTW that can be verified in polynomial time. Our characterization extends the approach developed by Bartolini et al. [14] for the single knapsack uncertainty set in the robust traveling salesman problem with time windows.

Consider a route $R = (r_0, r_1, \ldots, r_{m+1})$ visiting m customers, represented by the nodes r_p for $1 \leq p \leq m$, where r_0 and r_{m+1} correspond to the depot nodes 0 and $n+1$, respectively. Let $\tau^0_{r_p}$ denote the earliest possible service start time at node r_p in the deterministic setting, for $0 < p \leq m$, computed as: $\tau^0_{r_p} = \max\{w^a_{r_p}, \tau^0_{r_{p-1}} + s_{r_{p-1}} + \bar{t}_{r_{p-1}r_p}\}$. Note that the opening time of the time window may introduce waiting time. Additionally, define σ_{r_p} as the waiting time at node r_p in the deterministic setting, if any, computed as

$$\sigma_{r_p} = \tau^0_{r_p} - \tau^0_{r_{p-1}} - (s_{r_{p-1}} + \bar{t}_{r_{p-1}r_p}). \tag{12}$$

Let τ_{r_p} denote the earliest start time of service at node r_p in the robust problem. This value can be computed using the findings of the following theorem:

Theorem 3.1. *Let U be a closed, bounded, and convex uncertainty set, and let $\eta_{r_{i-1}r_i}$ denote the effective travel-time deviation induced on arc (r_{i-1}, r_i) by a realization $\eta \in U$. For a route $R = (r_0, \ldots, r_{m+1})$, a node r_p with $1 \leq p \leq m+1$, and an index $s \in \{0, \ldots, p-1\}$, let $\eta^{(s,p)}$ be a realization in U that maximizes the total deviation over the subpath $(r_s, \ldots, r_p)$, that is, $\sum_{i=s+1}^{p} \eta_{r_{i-1}r_i}$, and define*

$$\tau^{r_s}_{r_p} = \tau^0_{r_s} + \sum_{i=s+1}^{p} \left(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}} + \sigma_{r_i}\right) + \max_{0 \leq k \leq p} \left[(w^a_{r_k} - \tau^0_{r_k}) + \sum_{i=k+1}^{p} \left(\eta^{(s,p)}_{r_{i-1}r_i} - \sigma_{r_i}\right)\right], \tag{13}$$

whose first two terms equal the deterministic arrival $\tau^0_{r_p}$. Then the worst-case earliest service start time at r_p over all realizations in U is

$$\tau_{r_p} = \max\Big\{ w^a_{r_p}, \max_{0 \leq s \leq p-1} \tau^{r_s}_{r_p} \Big\}. \tag{14}$$

Proof. See Appendix A. □

Remark. *Although the realization $\eta^{(s,p)}$ maximizing the subpath deviation need not be unique, the value τ_{r_p} in (14) is independent of the choice: any such maximizer yields the same worst-case arrival time, as established in the proof.*

While the construction of Theorem 3.1 holds regardless of the uncertainty set, its verification simplifies for sets in which the deviation on each arc $(i,j) \in \mathcal{A}$ is governed by its own dedicated variable ξ_{ij}, ranging over a domain containing the nominal value $\xi_{ij} = 0$, and coupled with the other arcs only through constraints of the form $g(\xi) \leq b$, where g is nonnegative and nondecreasing in each $|\xi_{ij}|$ separately. We refer to such sets as *separable*. The cardinality-constrained, knapsack, and ellipsoidal sets all share this property, whereas in the factor-model and discrete sets, named *coupled* sets, a single component governs the deviations of several arcs simultaneously, ruling out this improvement.

For a separable set, the realization $\eta^{(s,p)}$ optimizing the subpath deviation can be extended by zero to every arc preceding r_s without affecting its optimality on $(r_s, \ldots, r_p)$, since arcs on either side of r_s share no component. Two features distinguish this case from the coupled one. First, since the deviation vanishes on all arcs preceding r_s, the inner maximization over k in (13) needs only account for the realization on the subpath $(r_s, \ldots, r_p)$ itself. Second, the outer maximization in (14) can be restricted to predecessors $s \in \mathbb{L} = \{\ell : \tau^0_{r_\ell} = w^a_{r_\ell}\}$, rather than ranging over all $s \in \{0, \ldots, p-1\}$. Both features are formalized in Proposition 1.

Proposition 1. *If, in addition to the assumptions of Theorem 3.1, U is separable, then for each arc the deviation is governed by its own component, and the realization optimizing the subpath $(r_s, \ldots, r_p)$ can be taken to be zero on every arc preceding r_s. Expression* (13) *can be replaced with*

$$\tau^{r_s}_{r_p} = \tau^0_{r_s} + \sum_{i=s+1}^{p} \left(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}}\right) + \max_{\eta \in U} \sum_{i=s+1}^{p} \eta_{r_{i-1}r_i}, \tag{15}$$

and the outer maximization in (14) *may be restricted to the predecessors $s \in \mathbb{L} = \{\ell : \tau^0_{r_\ell} = w^a_{r_\ell}\}$. In particular, the cardinality-constrained, knapsack, and ellipsoidal sets satisfy these conditions.*

Proof. See Appendix B. □

In a coupled set, by contrast, this extension is unavailable, since the component values that optimize the subpath $(r_s, \ldots, r_p)$ also fix the deviations on the preceding arcs. Consequently, the index attaining the maximum in (14) may satisfy $\tau^0_{r_s} \neq w^a_{r_s}$, so the outer maximization cannot be restricted to $\mathbb{L}$ as in the separable case, and must instead range over all predecessors $s \in \{0, \ldots, p-1\}$.

The proposed separation procedure, which relies on expressions (13)–(14) and (15), is summarized in Algorithm 1. Lines (1)–(5) compute the deterministic arrival time at each node visited in a route of the candidate solution. Robust feasibility is verified in lines (6)–(16), where the algorithm calculates the earliest time the service can start at each node r_p. In particular, line 10 updates this time (τ_{r_p}) by taking the maximum worst-case arrival time at node r_p when deviation is optimized on the sequence starting from a previous node r_s ($\tau^{r_s}_{r_p}$). The expression for $\tau^{r_s}_{r_p}$ depends on the chosen uncertainty set and is given in Table 1. Although the complexity for the factor-model and discrete sets grows with F and D, respectively, these parameters are typically much smaller than the number of nodes. As a result, their computational effort is effectively bounded by a polynomial in practical settings. Appendix D presents an illustrative example of the proposed approach using the ellipsoidal uncertainty set. The example also highlights why the worst-case arrival time cannot, in general, be obtained by simply maximizing the cumulative deviation along the route, as is commonly done in robust optimization problems under demand uncertainty.

For the factor-model and discrete uncertainty sets, the inner loop of Algorithm 1 is executed for every predecessor r_s, without the guard $\tau^0_{r_s} = w^a_{r_s}$ used in the separable case (Proposition 1). Moreover, once the realization optimizing the subpath $(r_s, \ldots, r_p)$ is determined, it is evaluated on the *entire* route in $\tau^{r_s}_{r_p}$: the inner maximization over $0 \leq k \leq p$ in (13) scans every node as a possible point from which the surviving delay accumulates, so that any deviation the shared components place on the arcs preceding r_s is correctly accounted for when it is not absorbed by an earlier buffer.

If τ_{r_p} exceeds the time-window closing time $w^b_{r_p}$, the route is deemed infeasible, and the corresponding solution is rejected. The algorithm proceeds by repeating this process for each route. At each iteration, either a cut is generated for every infeasible route, or the solution is accepted as robustly feasible.

Since the algorithm repeatedly computes the maximum possible deviation along a subpath, it is essential to solve the associated maximization subproblem efficiently for each uncertainty set. To this end, we propose closed-form expressions that compute this value in polynomial time. These expressions are summarized in Table 1, which presents the formulas for the arrival time at node r_p for each uncertainty set, assuming that deviation occur only along a subpath starting from index $s < p$ (denoted $\tau^{r_s}_{r_p}$). For clarity, we introduce the following notation used in the formulas:

Algorithm 1: Feasibility evaluation procedure

Input: Solution candidate.
Output: Information on the feasibility of the solution for the uncertainty set U.

foreach *route* $R = (r_0, r_1, \ldots, r_{m+1})$ **do**
  $\tau^0_0 \leftarrow w^a_0$;
  for $p \leftarrow 1$ **to** $m+1$, *step* $+1$ **do**
    $\tau^0_{r_p} \leftarrow \max\{w^a_{r_p}, \tau^0_{r_{p-1}} + \bar{t}_{r_{p-1}r_p} + s_{r_{p-1}}\}$;
  end
  for $p \leftarrow 1$ **to** $m+1$, *step* $+1$ **do**
    $\tau_{r_p} \leftarrow \tau^0_{r_p}$;
    for $s \leftarrow (p-1)$ **to** 0, *step* -1 **do**
      if U *is separable* **and** $\tau^0_{r_s} = w^a_{r_s}$, **or** U *is coupled* **then**
        $\tau_{r_p} \leftarrow \max\{\tau_{r_p}, \tau^{r_s}_{r_p}\}$;
        if $\tau_{r_p} > w^b_{r_p}$ **then**
          return *Solution is infeasible;*
        end
      end
    end
  end
end
return *Solution is feasible for the evaluated uncertainty set;*

- S_l: subset of arcs in knapsack l.
- Ps: number of arcs in the subpath, computed as $Ps = p - s$.
- $\hat{t}_{(i)}$: the deviations along the subpath, indexed in non-increasing order.
- f^+ and f^-: indices used in the closed-form expressions for the factor model, defined as $f^+ = \left\lceil \frac{(1+\beta)F}{2} \right\rceil$ and $f^- = \max\left\{ \left\lceil \frac{(1-\beta)F}{2} \right\rceil, 1 \right\}$.
- $\Psi^{(s,p)}_f := \sum_{i=s+1}^{p} \Psi_{r_{i-1}r_i f}$, $f = 1, \ldots, F$: aggregate factor loading along the subpath $(r_s \ldots r_p)$ for the factor-model uncertainty set. We assume that the values of the loading matrix Ψ for each factor are sorted in non-increasing order, so that $\Psi^{(s,p)}_1 \geq \cdots \geq \Psi^{(s,p)}_F$.
- σ_{r_i}: slack at node r_i resulting from the time-window opening constraint in the deterministic case;
- $\xi^{(s,p)}_f$: worst-case realization for the subpath $(r_s \ldots r_p)$, given by $\xi^{(s,p)}_f = \text{sign}(\Psi^{(s,p)}_f - \lambda^*)$ for $\Psi^{(s,p)}_f \neq \lambda^*$, where λ^* is the optimal breakpoint for the closed-form expression of the factor model uncertainty set, while the factors tied at λ^* split the residual budget equally (see Corollary C.4.1 in Appendix C).
- $\delta^{(s,p)}_{r_{i-1}r_i} = \sum_{f=1}^{F} \xi^{(s,p)}_f \Psi_{r_{i-1}r_i f}$: deviation induced on arc (r_{i-1}, r_i) by the subpath-optimal realization $\xi^{(s,p)}$.
- $\varsigma^{(s,p)}$: index of the scenario realization with highest total deviation for the subpath $(r_s \ldots r_p)$, i.e. $\varsigma^{(s,p)} \in \arg\max_{j \in \{1,\ldots,D\}} \sum_{i=s+1}^{p} \hat{t}^{(j)}_{r_{i-1}r_i}$.

The coupled rows (factor model and discrete) carry the additional term $w^a_{r_k} - \tau^0_{r_k} \leq 0$ inside the inner maximization. This term is the slack between the time-window opening and the deterministic service start at the candidate reset node r_k, and it accounts for realizations in which a negative deviation makes the vehicle

reach r_k before $\tau^0_{r_k}$, so that service starts at the window opening rather than on the deterministic schedule. The term vanishes at nodes with $\tau^0_{r_k} = w^a_{r_k}$, and in particular at every node for the separable rows, whose nonnegative deviations never advance an arrival; it is therefore omitted there.

Furthermore, Table 1 reports the computational complexity of this calculation and the overall algorithm for each uncertainty set. The derivation of each closed-form expression is provided in Appendix C.

Table 1: Mathematical expression and computational complexity used in the proposed algorithm

Uncertainty set	**Closed-Form**	**Computational Complexity**	
		Single verification	**Algorithm**
Cardinality-constrained (U_Γ)	$\tau^{r_s}_{r_p} = \tau^0_{r_s} + \sum_{i=s+1}^{p}(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}}) + \sum_{i=1}^{\min\{Ps,\lfloor\Gamma\rfloor\}} \hat{t}_{(i)} + \mathbf{1}_{\Gamma<Ps}(\Gamma - \lfloor\Gamma\rfloor)\hat{t}_{(\lfloor\Gamma\rfloor+1)}$	$n \log n$	$n^3 \log n$
Knapsack (U_Δ)	$\tau^{r_s}_{r_p} = \tau^0_{r_s} + \sum_{i=s+1}^{p}(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}}) + \sum_{l\in L} \min\left\{\Delta_l, \sum^{p}_{\substack{i=s+1 \\ (r_{i-1},r_i)\in S_l}} \hat{t}_{r_{i-1}r_i}\right\}$	n	n^3
Ellipsoidal (U_Ω)	$\tau^{r_s}_{r_p} = \tau^0_{r_s} + \sum_{i=s+1}^{p}(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}}) + \Omega\sqrt{\sum_{i=s+1}^{p}(\hat{t}_{r_{i-1}r_i})^2}$	n	n^3
Factor model (U_Ψ)	$\tau^{r_s}_{r_p} = \tau^0_{r_s} + \sum_{i=s+1}^{p}\left(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}} + \sigma_{r_i}\right) + \max_{0\le k\le p}\left[(w^a_{r_k} - \tau^0_{r_k}) + \sum_{i=k+1}^{p}\left(\delta^{(s,p)}_{r_{i-1}r_i} - \sigma_{r_i}\right)\right]$	nF	n^3F
Discrete (U_D)	$\tau^{r_s}_{r_p} = \tau^0_{r_s} + \sum_{i=s+1}^{p}\left(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}} + \sigma_{r_i}\right) + \max_{0\le k\le p}\left[(w^a_{r_k} - \tau^0_{r_k}) + \sum_{i=k+1}^{p}\left(\hat{t}^{(\varsigma^{(s,p)})}_{r_{i-1}r_i} - \sigma_{r_i}\right)\right]$	nD	n^3D

4 Branch-and-cut method

To solve the RVRPTW we first implemented a BC method using the proposed algorithm. This method makes use of cuts developed for deterministic variants of the studied problems and strategies specifically tailored to deal with the RO problems. The cuts designed for the deterministic problem are separated using the CVRPSep package [25].

Three RO-based strategies are used in the proposed approach. First, we introduce a robust adaptation of the classical time-window tightening and arc elimination preprocessing. This adaptation explicitly exploits the structure of the uncertainty set and the associated deviation information, resulting in tighter bounds compared to the traditional deterministic procedure. The second strategy is to use the separation algorithm described in Section 3 and introduce no-good cuts if a route is found to be infeasible. This step is crucial to ensure that the final solution obtained by the solution method is robustly feasible. Finally, we use the information about the deviation to propose robust reachability cuts (R-cuts) adapted for each uncertainty set. These cuts lead to stronger relaxation than their deterministic counterpart, as they account for deviation along the routes. To the best of our knowledge, we are the first to develop and integrate robust reachability cuts for the RVRPTW. In the remainder of this section, we describe the separation algorithm, the strategies used to improve its performance, and how RO is incorporated into the procedure.

4.1 Robust graph preprocessing.

The first step of the BC method is a preprocessing phase, in which we tighten the time windows and remove arcs that cannot appear in any feasible solution. This is done by extending the standard preprocessing rules developed for the deterministic problem [26], along with tailored rules for the robust counterpart [14]. We assume that the triangular inequality holds for the nominal and maximum travel times of each arc, i.e., $\bar{t}_{ij} \leq \bar{t}_{ik} + \bar{t}_{kj}$ and $\bar{t}_{ij} + \hat{t}_{ij} \leq (\bar{t}_{ik} + \hat{t}_{ik}) + (\bar{t}_{kj} + \hat{t}_{kj})$ for all $i, j, k \in \mathcal{N}$.

The robust procedure follows the same general structure as its deterministic counterpart, replacing each travel time with its worst-case value under the chosen uncertainty set U, captured through the closed-form deviation parameter α_{ij} derived from the formulas of Table 1. This substitution requires some care, however, as its effect is not uniform across the improvement strategies. Arc removal can become more effective than in the deterministic case, since worst-case deviations may render additional arcs infeasible beyond those already excluded deterministically. Conversely, time-window tightening can become less effective in some directions, as a bound that holds under the deterministic travel time need not hold under its worst-case realization, so the corresponding window can only be tightened by a smaller margin, if at all. The complete arc-removal and time-window-tightening procedure is detailed in Appendix E.

4.2 No-good cuts.

Whenever a new solution of the LP relaxation is found, the solver invokes the separation algorithm discussed in Section 3. If the solution is fractional, rounding strategies are used to generate routes to be evaluated. If the LP relaxation contains an integer solution that is robust feasible, the solver accepts it as a candidate for optimal solution. Conversely, if the solution is found infeasible by the separation algorithm in route $R = (r_0, \ldots, r_p, \ldots, r_{m+1})$ at node r_p, the solver rejects the solution by adding the following no-good cut to the model:

$$x_{r_0 r_1} + x_{r_1 r_2} + \cdots + x_{r_{p-1} r_p} \leq p - 1. \tag{16}$$

It is possible to strengthen these cuts by identifying the latest predecessor of r_p in node sequence $(r_0, \ldots, r_p)$, r_s, from which the deviation can start to be considered such that the route becomes infeasible in node r_p. From Algorithm 1, this node r_s is the latest predecessor whose subpath would result in $\tau_{r_p} > w^b_{r_p}$. With this information, constraint (16) can be strengthened with the following expression:

$$x_{r_s r_{s+1}} + \cdots + x_{r_{p-1} r_p} \leq (p - s) - 1. \tag{17}$$

Expression (17) can be further strengthened using *tournament cuts* [26]. The central idea is that if the subpath $R' = (r_s, \ldots, r_p)$ is infeasible and the travel times satisfy triangle inequality (which is already assumed in Section 4.1), then any other path that visits a subset of these nodes in the same order is also infeasible. To exploit this property, we collect every arc that respects the order of R' into the set

$$\zeta(r_s, r_p) \;=\; \big\{\, (r_i, r_j) \;:\; s \leq i < j \leq p \,\big\}.$$

This set contains the $p - s$ arcs of the original subpath together with every additional arc (r_i, r_j) that connects two non-consecutive nodes of R' while preserving their order. We partition $\zeta(r_s, r_p)$ into three subsets according to which start and endpoint of R' is involved:

$$\zeta(r_s, r_p) \;=\; \zeta'(r_s, r_p) \,\cup\, \zeta''(r_s, r_p) \,\cup\, \zeta'''(r_s, r_p),$$

where ζ' contains the arcs leaving r_s, ζ'' the arcs between internal nodes, and ζ''' the arcs entering r_p.

Depots require special treatment. When $r_s = 0$, an arc (r_0, r_j) with $j > 1$ is not necessarily infeasible: in another solution, it could be the first arc of a different route that does not visit $r_1, \ldots, r_{j-1}$ beforehand.

Hence, including these arcs in ζ' would risk cutting off feasible solutions. We thus keep only the original arc (r_0, r_1) in ζ'. A symmetric argument applies to arcs entering the sink depot $r_p = n + 1$. Formally,

$$\zeta'(r_s, r_p) = \begin{cases} \{(r_s, r_{s+1})\}, & \text{if } r_s = 0, \\ \big\{(r_s, r_j) : j = s+1, \dots, p\big\}, & \text{otherwise,} \end{cases}$$

$$\zeta''(r_s, r_p) = \big\{(r_i, r_j) : s+1 \le i < j \le p-1\big\},$$

$$\zeta'''(r_s, r_p) = \begin{cases} \{(r_{p-1}, r_p)\}, & \text{if } r_p = n+1, \\ \big\{(r_i, r_p) : i = s+1, \dots, p-1\big\}, & \text{otherwise.} \end{cases}$$

The tournament cut associated with the infeasible subpath R' is then

$$\sum_{(i,j) \in \zeta(r_s, r_p)} x_{ij} \;\le\; (p - s) - 1. \tag{18}$$

The right-hand side is the number of arcs in the original subpath minus one, so the cut forbids any selection of $p - s$ order-preserving arcs over R'. It strictly dominates the no-good cut (17), which only forbids the specific sequence of consecutive arcs in R'.

4.3 Robust R-cuts.

We further extend the reachability cuts (R-cuts) of Lysgaard [27] to account for travel-time uncertainty and implement them within our framework. As in the deterministic case, these cuts remove arcs that cannot lie on a feasible path connecting the depot to a given customer, but the robust variant evaluates feasibility using the worst-case path duration under the uncertainty set U, computed via the preprocessing step of Section 4.1, rather than the deterministic travel time. Full details are provided in Appendix F.

5 Branch-price-and-cut method

BPC is currently the most effective exact method for solving the RCVRP [11] and the RVRPTW [6]. For this reason, we implement a BPC method that uses our proposed separation algorithm to solve the RVRPTW. The proposed approach is an adaptation of the robust cutting-plane approach developed by Wang et al. [11] using the BPC from the VRPSolver [12] as a basis. The remainder of this section provides an overview of its main components.

5.1 Set partitioning formulation

The BPC method is based on a set partitioning formulation of the problem. In this type of formulation, instead of explicitly defining the routes of the vehicle, we are given a set of possible routes as input, and the goal of the model is to decide which routes to take among those available. Let $\mathcal{R}$ be the set of all robust feasible routes for the RVRPTW, c_r be a parameter representing the cost associated with route $r \in \mathcal{R}$, and a_{ir} be a binary parameter that takes the value 1 if node i is visited on route $r \in \mathcal{R}$. Additionally, we define a binary decision variable λ_r which is 1 if route r is used in the solution and 0 otherwise. With this, the RVRPTW can be modeled as:

$$\min \sum_{r \in \mathcal{R}} c_r \lambda_r \tag{19}$$

$$\text{s.t.} \sum_{r \in \mathcal{R}} a_{ir} \lambda_r = 1, \qquad i \in \mathcal{N}^*, \tag{20}$$

$$\sum_{r \in \mathcal{R}} \lambda_r \leq V, \tag{21}$$

$$\lambda_r \in \{0,1\}, \qquad r \in \mathcal{R}. \tag{22}$$

The objective function (19) minimizes the total routing cost. Constraints (20) ensure that each customer is visited exactly once, while constraint (21) restricts the number of routes to the available fleet size V. Finally, constraints (22) define the domains of the decision variables.

Enumerating all possible feasible routes and inserting them into the model is impractical due to the prohibitively large number of routes. Therefore, it would take a significant amount of time and result in an excessive number of unused variables. To address this issue, CG algorithms are commonly used to generate a subset of promising routes to be inserted into the model. The set of routes in this reduced master problem is denoted by $\tilde{\mathcal{R}}$. The proposed CG algorithm is described in the next topic.

The combination of the CG algorithm with a branch-and-bound tree results in the so-called branch-price method. Since the proposed method also uses cutting-plane procedures, it is called a BPC.

5.2 Robust cutting-plane approach

We ensure the robust feasibility of the solution by using a cutting plane strategy. Whenever the method finds a candidate optimal solution, we use the separation procedure described in Section 3 to evaluate if the columns used in the current solution are robust feasible. If the solution is infeasible due to a subpath $R' = (r_s, \ldots, r_p)$ within route $R = (r_0, r_1, \ldots, r_s \ldots, r_p, \ldots, r_{m+1})$, we add the tournament cuts to the RMP:

$$\sum_{r \in \tilde{\mathcal{R}}} \omega_r \lambda_r \leq (p - s) - 1, \tag{23}$$

where parameter ω_r represents the number of arcs in route $r \in \mathcal{R}$ that are within subset $\zeta(r_s, r_p)$, the subset of arcs on the left-hand side of the tournament cut associated to R'. When a new column is generated, the parameter ω_r is computed for each cut in the RMP. The only significant modifications to the labeling algorithm are: incorporating the dual costs associated with the added cuts for each prospective column label, and disabling the enumeration procedure that eliminates "similar" routes, i.e., routes that visit the same customers in different sequences. This is because a route that appears cost-effective may later be identified as not robust feasible.

It is worth noting that these adjustments do not increase the complexity of solving the pricing subproblem [11]. This also highlights the modularity of the proposed separation algorithm, which can be easily integrated into different solution methods.

5.3 Graph construction

While it is possible to construct the graph for the subproblem in the CG algorithm using deterministic travel times and demands, this approach can be inefficient, as the labeling algorithm is more likely to generate columns that will ultimately be eliminated by the cutting-plane algorithm. For this reason, it is advantageous to extend the graph to account for uncertainty. An interesting strategy that balances the increase in complexity of solving the subproblem with the generation of more relevant columns is to produce a subset of scenarios for the uncertain parameters within the uncertainty set [11]. This strategy can significantly improve the algorithm's overall performance by filtering out routes that are not robustly feasible at the CG stage. When well-chosen scenarios are incorporated into the labeling algorithm, a large proportion of infeasible routes are never explored. Consequently, fewer infeasible columns are generated, which reduces the number of separation calls and no-good cuts required compared to a purely deterministic graph. Notably, this improvement is achieved with a limited increase in pricing complexity. In fact, the main difference in terms of complexity of the labeling algorithm, compared to the deterministic case, lies in the addition of extra resources that must be extended and evaluated in the dominance rule for each scenario.

For each uncertainty set U considered in this work, we design a tailored subset $\mathcal{T}_U$ aimed at generating useful scenarios that reduce the effort required to separate necessary constraints through the cutting-plane algorithm. These subsets are constructed as follows:

- **Cardinality-constrained set ($\mathcal{T}_{U_\Gamma}$)**: For the cardinality-constrained uncertainty set, we construct a list $\mathcal{T}_{U_\Gamma}$ of $\lceil\Gamma\rceil$ nodes for each scenario $\mathcal{T}^i_{U_\Gamma}$. Each list $\mathcal{T}^i_{U_\Gamma}$ starts with the depot, and its second element is the i^{th} closest customer to the depot. The remaining elements are selected as the closest customer to the preceding one in the list. If two lists end up containing the same customer, we restart one of the conflicting lists using the next closest customer to the depot that has not yet been placed in the first position of any other list. Each scenario is then generated as follows: for the nodes in the first $\lfloor\Gamma\rfloor$ positions of the list $\mathcal{T}_{U_\Gamma}$, we assume that all arcs departing from those nodes attain their worst-case values. If Γ is fractional, all arcs departing from the node in position $\lceil\Gamma\rceil$ are assigned a deviation equal to $(\Gamma - \lfloor\Gamma\rfloor)\hat{t}_{ij}$. This construction is valid because each route must be feasible under any realization in which at most Γ arcs experience their worst-case deviation, and in the worst case in these scenarios, a single vehicle can traverse at most Γ such arcs, as each customer is only visited once. We note that the scenarios proposed here are stronger than those in Wang et al. [11], which only included $\lceil\Gamma\rceil$ arcs with deviations per scenario.

- **Knapsack set ($\mathcal{T}_{U_\Delta}$):** For this set, each scenario $\mathcal{T}^i_{U_\Delta}$ is associated with two nodes, denoted by $\{i_1, i_2\}$. The first node, i_1, is randomly selected, and the second node, i_2, is the closest customer to i_1 such that the arc (i_1, i_2) was not removed during the arc removal procedure, if no such arc exists, the scenario is skipped. The scenario is then constructed as follows. All arcs departing from i_1 attain their worst-case travel time values, that is, $t_{i_1 j} = \bar{t}_{i_1 j} + \min\{\hat{t}_{i_1 j}, \Delta_{l(i_1,j)}\}$, $\forall j \in \mathcal{N}^*$, where $l(i_1, j)$ denotes the index of the knapsack containing arc (i_1, j). Then, all arcs departing from the second node i_2 are also delayed, using the assumption that arc (i_1, i_2) was taken. Thus, for a given node j, the travel time between i_2 and j depends on whether arcs (i_1, i_2) and (i_2, j) belong to the same knapsack. If $l(i_1, i_2) \neq l(i_2, j)$, then $t_{i_2 j} = \bar{t}_{i_2 j} + \min\{\hat{t}_{i_2 j}, \Delta_{l(i_2,j)}\}$. Otherwise, if $l(i_1, i_2) = l(i_2, j)$, then $t_{i_2 j} = \bar{t}_{i_2 j} + \min\left\{\hat{t}_{i_2 j}, \max\left\{\Delta_{l(i_2,j)} - \hat{t}_{i_1 i_2}, 0\right\}\right\}$.

- **Ellipsoidal set ($\mathcal{T}_{U_\Omega}$):** In the axis-parallel ellipsoidal uncertainty set, each scenario $\mathcal{T}^i_{U_\Omega}$ is associated with a randomly selected node i in $\mathcal{N}^* \cup \{0\}$. In each scenario, all arcs departing from the selected node are assumed to attain the value $\bar{t}_{ij} + \Omega\hat{t}_{ij}$, $j \in N\backslash\{0\}$, the worst-case realization achievable with budget Ω.

- **Factor model set ($\mathcal{T}_{U_\Psi}$):** For the factor model, we generate scenarios by modifying the variables ξ_f, which represent the magnitudes of the factors f. The procedure is as follows: if the budget $\beta F = 0$, we select pairs of factors (f_1, f_2) and set $\xi_{f_1} = 1$ and $\xi_{f_2} = -1$, while all other factors are assigned $\xi_f = 0$. If $\beta F > 0$, we first generate scenarios where a single factor f is set to $\xi_f = \beta F$, and the remaining factors are set to zero. Since the instances considered in this work include four factors, we also generate additional scenarios involving combinations of three factors (f_1, f_2, f_3), setting $\xi_{f_1} = 1$, $\xi_{f_2} = -1$, and $\xi_{f_3} = \beta F$. Notably, in our computational experiments, the budgets βF were limited to 1. For larger values, it suffices to set $\xi_f = \min\{1, \beta F\}$.

- **Discrete set ($\mathcal{T}_{U_D}$):** In this uncertainty set, we replicate the procedure proposed by Wang et al. [11]. We first eliminate dominated scenarios, i.e., any scenario s such that $t^s_{ij} \leq t^{s'}_{ij}$ for all $(i, j) \in \mathcal{A}$, for some scenario $s' \neq s$. Then, we sort the remaining scenarios by their total arc traversal time in non-increasing order and select the first $|\mathcal{T}_{U_D}|$ scenarios to be added as additional resources in the graph.

The number of scenarios in each set can be empirically tuned to balance pricing subproblem complexity and the number of constraints generated, with the goal of improving the overall performance of the method.

5.4 Initial solution

An effective initial solution can accelerate the convergence of the BPC method by providing a feasible set of routes to warm-start the master problem and reducing the number of unpromising labels generated. In our implementation, we generate initial columns using two simple constructive heuristics and a local search algorithm, executed in parallel. Throughout all heuristic procedures, time-window feasibility is verified using our separation procedure described in Algorithm 1. The first constructive heuristic is a route-merging heuristic based on the classical Clarke–Wright savings method. In this algorithm, each customer is initially assigned to its own route, and we repeatedly merge the pair of routes that yields the greatest feasible savings (i.e., the cheapest combined route) until no further merges are possible.

The second constructive heuristic is a greedy insertion heuristic. We first estimate the number of vehicles to be used in the solution by dividing the total demand by the vehicle capacity, and then build that many routes in parallel. Customers are inserted into each route in turn: in each iteration, the cheapest feasible insertion of an unvisited customer is performed. If some customers cannot be inserted into any of the initial routes, a new route is created, and the greedy insertion process continues until all customers have been assigned.

To improve the quality of the initial heuristic solutions, each of the two initial solutions is refined using a local search algorithm. We consider four move types, applied sequentially: single-customer relocation (moving one customer to another position within the same route or to a different route); double-customer relocation (moving two consecutive customers together to another position); customer swap (exchanging two customers between routes); and 2-opt (reversing a segment within a route).

We run this algorithm twice in parallel, using two strategies: first-improvement and best-improvement. In both cases, each move type tests every possible customer insertion. The main difference is that, in the first-improvement strategy, the search in a particular move stops immediately after a feasible improvement is found, whereas in the best-improvement strategy, all possible reassignments or swaps are evaluated, and the most economical one is selected. Whenever a move yields a feasible improved solution, we update the current solution. The search terminates when no improvement is found. The pseudocode presented in Algorithm 2 summarizes the local-improvement phase. The *applyMove* function in this algorithm describes the process of checking the solution by a specific move and returns either an improved feasible solution, based on the strategy chosen between first- and best-improvement, or the current solution a' with no changes.

Algorithm 2: Local-search improvement procedure.

Input: Feasible solution a, local-search *strategy* (first improve, best improve).
Output: Locally improved solution a'.

$a' \leftarrow a$
$improved \leftarrow true$
while *improved* **do**
  $improved \leftarrow false$
  foreach *move* $\in$ {*1-relocate*, *2-relocate*, *swap*, *2-opt*} **do** `// apply moves in sequence`
    $a'' \leftarrow$ applyMove(a', move, strategy)
    if $f(a'') < f(a')$ **then**
      $a' \leftarrow a''$
      $improved \leftarrow true$
    end
  end
end
return a'

6 Computational results

This section presents the results of the computational study designed to evaluate the performance of proposed robust solution approaches for the RVRPTW. We implement the BC and BPC methods, as described in Sections 4 and 5, and assess their effectiveness under the uncertainty sets presented in Subsection 2.2. Additionally, these experiments analyze the impact of incorporating robustness into the solutions, as well as the resulting trade-offs in terms of solution reliability and the resulting objective function value.

In this computational study, we use benchmark instances from the RVRPTW literature. We adapted the Solomon's instances [28], originally designed for the VRPTW, to include uncertainty on travel times similarly to other works in the RVRPTW literature [6]. It consists of 56 instances divided into six classes: C1, C2, R1, R2, RC1, and RC2. Classes C1 and C2 are characterized by having the customers distributed in clusters, R1 and R2 have the nodes randomly positioned, and classes RC1 and RC2 are a mix of random and clustered structures. Moreover, problem sets C1, R1, and RC1 differ from C2, R2, and RC2 by having smaller capacities and shorter planning horizons, which results in fewer customers per route. To incorporate uncertainty into the instances, we defined $\hat{t}_{ij}$ as 10%, 25%, and 50% of the nominal travel time (original value for the instance), truncated to one decimal place. The remaining parameters, such as costs and time windows, are identical to those in the corresponding deterministic instances.

These instances were tested in each uncertainty set considering different budgets. For the cardinality-constrained uncertainty set, we considered the same budget Γ and deviation levels used by Munari et al. [6], i.e., $\Gamma = 1,\ 5,$ and 10.

To generate the instances with knapsack and factor-model uncertainty sets, we follow the strategy adopted in the robust CVRP literature [5, 18], in which the nodes are partitioned into four geographic quadrants (NE, SE, NW, and SW) according to their coordinates. Each quadrant defines a knapsack and a factor. For the knapsack set, each arc (i, j) is assigned to the knapsack of its destination node's quadrant. For the factor model, each arc loads on all four factors through $\Psi_{ijf} = \dfrac{\hat{t}_{ij}\,\psi_{jf}}{\sum_{f'=1}^{F}\psi_{jf'}}$, where ψ_{jf} is the inverse of the distance between customer j and the centroid of quadrant $f \in \{\text{NE}, \text{SE}, \text{NW}, \text{SW}\}$. We set $\Psi' := \beta F$ and, with $F = 4$, consider budgets $\Delta = 20,\ 40,$ and 60 for the knapsack set and $\Psi' = 0,\ 0.25,\ 0.5,\ 0.75,$ and 1 for the factor model.

For the axis-parallel ellipsoidal uncertainty set we also consider budgets $\Omega = 0.25$, 0.5, 0.75 and 1. Finally, for the discrete uncertainty set, we test the instances with different number of scenarios $D = 10,\ 20,\ 30,$ and 40. Each scenario is randomly generated considering a uniform distribution in which the travel time between two nodes attains a value ranging from $\bar{t}_{ij}$ to $\bar{t}_{ij} + \hat{t}_{ij}$ in each scenario. The information on these budgets is summarized in Table 2.

The remainder of this section is structured as follows. In Subsection 6.1, we report the computational results of the solution methods. In Subsection 6.2, we present a robustness analysis of the solutions, highlighting the trade-offs between solution cost and infeasibility risk. Finally, in Subsection 6.3, we examine the structure of individual solutions under each uncertainty set with increasing budgets.

All experiments were carried out on a computer cluster with nodes equipped with an AMD Rome 7532 @ 2.40 GHz 256M processor, using 32 GB of memory and a time limit of 3600 seconds. Up to 32 threads were

Table 2: Parameters for each uncertainty set used in the computational experiments

Uncertainty set	**Parameter**	**Value**
Cardinality-constrained (U_Γ)	Γ	1, 5 and 10
Knapsack (U_Δ)	Δ	20, 40 and 60
Ellipsoidal (U_Ω)	Ω	0.25, 0.5, 0.75 and 1
Factor model (U_Ψ)	Ψ'	0, 0.25, 0.5, 0.75 and 1
Discrete (U_D)	D	10, 20, 30 and 40

available for each experiment. The BC method was coded in C++ using the Concert library of IBM CPLEX Optimization Studio version 22.1 with default parameters. The BPC method was implemented through the C++ interface of the VRPSolver within Bapcod (version 0.83.1). Except for disabling the enumeration procedure, which is not compatible with the robust algorithm, all default parameters from the deterministic VRPTW demo provided with the solver were used. In the graph construction step described in Subsection 5.3, we created 10 resources for each instance.

6.1 Computational performance

In this section, we assess the computational performance of the BPC approach for each uncertainty set. We focus on this algorithm because it consistently outperforms the BC method described in Section 4. For completeness, the computational results obtained with the BC method are reported in Appendix G.

The results obtained using the implemented BPC method are summarized in Tables 3 and 4. Table 3 reports, for each instance size in number of customers (nN), the number of instances tested (Ins), the number of instances solved to optimality or proven infeasible (nS), the average computing time in seconds (T), the average optimality gap (Gap) returned by the solver at termination, and the number of capacity (Capacity) and tournament (Tourn) cuts generated throughout the execution of the solution algorithm. Table 4 provides a more detailed breakdown of computing times, reporting the average BPC solution time for each combination of budget and deviation level within each uncertainty set studied.

Overall, the BPC method demonstrates strong and consistent performance across the tested instance sizes. Small instances (25 customers) are solved to optimality in nearly all cases, with success rates above 95% for every uncertainty set. Medium-sized instances (50 customers) are also solved reliably, with optimality reached in 69.4% to 83.5% of cases depending on the uncertainty set. For the largest instances tested (100 customers), the method solves between 27.0% and 46.1% of instances to optimality; for those that remain open at the time limit, the average optimality gaps stay relatively small, ranging from 5.7% to 8.8%, indicating that the method still produces high-quality solutions even when optimality cannot be certified.

Although this approach does not outperform the state-of-the-art method of Munari et al. [6] for the cardinality-constrained set, our proposed method is considerably easier to implement. It is simpler to integrate the separation procedure into a framework for the deterministic problem than to introduce the substantial modifications required to generate robust feasible columns efficiently. Furthermore, our approach can be applied to a broader range of uncertainty sets, as it can be used under mild assumptions.

On average, the robust counterparts require more time than the deterministic problem. This behavior was expected, as the method for the robust counterpart applies the separation procedure on top of the framework for the deterministic problem. Consequently, in the best-case scenario, when the optimal solution of the deterministic problem is already robust feasible, the method for the robust problem takes the same amount of time as the method for the deterministic problem, plus the time required for a single verification. In most cases, however, the optimal solution of the deterministic problem is rejected, and additional time is needed to identify and verify new candidate solutions. Notably, this increase in computing time remained relatively small, with most configurations showing increases of less than 400 seconds and all configurations requiring less than twice the deterministic time. Additionally, the optimality gaps for instances not solved to optimality were similar in both the deterministic and robust settings.

When analyzing the impact of budget size on computational performance, we note that, in general, larger budgets for a given uncertainty set lead to longer computing times. This occurs because the method must explore and remove more solutions in more conservative configurations. This trend is observable across all uncertainty sets: for a fixed value of estimated deviation (Dev), computing time generally increases as the budget grows. In some configurations, however, this trend is not strictly followed. For example, when comparing average times for the factor model uncertainty set with budgets $\Psi' = 0.75$ and 1 at maximum deviations of 25% or 50% over the nominal, instances with $\Psi' = 1$ actually require less time. In most cases, this is because, as seen in Table 5, which reports the number of infeasible instances per configuration of budget and deviation, more conservative configurations lead to a higher number of instances deemed

Table 3: Computational results of the BPC solver for the RVRPTW instances

			Solved				Cuts	
Uncertainty set	nN	Ins	nS	nS(%)	T(s)	Gap(%)	Capacity	Tourn
Deterministic	25	56	56	100.0	46.2	0.0	38.5	0.0
	50	56	50	89.3	497.1	0.9	185.9	0.0
	100	56	27	48.2	2144.3	7.3	2640.5	0.0
Cardinality	25	504	497	98.6	88.5	0.2	5120.0	50.7
	50	504	391	77.6	1038.8	2.5	34430.3	1149.8
	100	504	164	32.5	2750.8	7.2	23244.9	1219.1
Knapsack	25	504	483	95.8	204.1	0.7	10770.9	442.4
	50	504	350	69.4	1303.3	3.9	39431.4	2464.6
	100	504	136	27.0	2892.2	8.8	20948.5	1851.0
Ellipsoidal	25	672	671	99.9	27.9	0.0	343.8	4.5
	50	672	561	83.5	808.8	1.6	25625.1	458.0
	100	672	247	36.8	2602.4	5.7	29589.9	741.9
Factor Model	25	840	838	99.8	32.5	0.0	3228.1	8.8
	50	840	676	80.5	942.4	2.9	21237.4	146.5
	100	840	387	46.1	2405.5	8.6	9634.7	145.6
Discrete	25	672	645	96.0	175.5	0.0	216.8	1.0
	50	672	482	71.7	1178.8	3.7	14341.1	108.1
	100	672	210	31.3	2730.2	8.8	12816.7	149.1

Table 4: Average computing times of the solutions obtained for each uncertainty set considering different deviation levels in the RVRPTW instances.

	Time(s)									
	Det	U_Ψ					U_D			
Dev	-	0	0.25	0.5	0.75	1	10	20	30	40
10%	900.9	1017.8	1108.7	1186.9	1055.0	1119.2	1010.7	1158.3	1275.2	1254.7
25%	900.9	1018.0	1127.7	1221.7	1190.3	1168.1	1095.6	1459.2	1523.5	1488.3
50%	900.9	1130.2	1204.5	1144.7	1157.0	1152.8	1207.7	1592.0	1603.0	1669.2
	U_Γ			U_Δ			U_Ω			
Dev	1	5	10	20	40	60	0.25	0.5	0.75	1
10%	1057.3	1197.3	1179.4	1388.9	1349.7	1329.5	999.1	1076.2	984.7	1139.0
25%	1140.1	1400.1	1414.5	1441.5	1530.7	1443.0	983.0	1121.6	1272.9	1216.6
50%	1286.8	1519.5	1439.1	1511.3	1583.5	1620.5	1119.1	1229.6	1254.9	1285.8

infeasible, which typically reduces computational time, since proving infeasibility is considerably faster than obtaining and verifying an optimal solution. Nevertheless, the overall increase in time with higher budgets is modest. The average difference between the shortest and longest times across budgets (for fixed deviation and uncertainty set) ranges from only 59.37 seconds to 461.57 seconds.

Table 5: Number of infeasible RVRPTW instances for each configuration of budget and deviation level

	Infeasible									
	Det	U_Ψ					U_D			
Dev	-	0	0.25	0.5	0.75	1	10	20	30	40
10%	0	0	0	0	0	0	0	0	0	0
25%	0	0	0	0	0	8	0	1	1	1
50%	0	0	6	12	12	15	5	8	9	10
	U_Γ			U_Δ			U_Ω			
Dev	1	5	10	20	40	60	0.25	0.5	0.75	1
10%	0	0	0	0	0	1	0	0	0	0
25%	7	7	7	7	7	7	0	0	0	7
50%	15	15	15	15	15	15	0	7	11	15

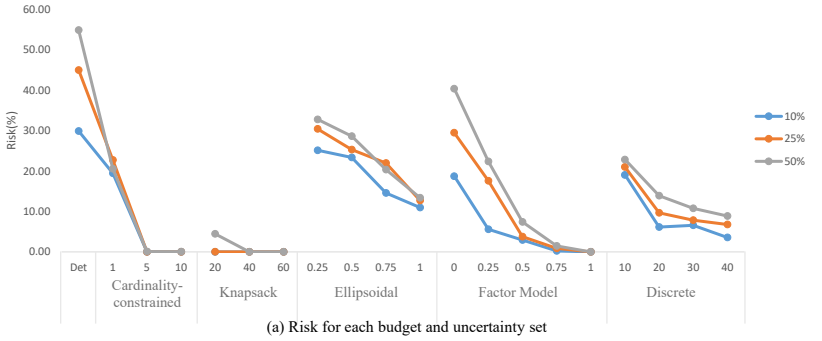
A similar pattern can be observed when evaluating the effect of worst-case travel-time deviations on the method's computational performance. As the problem considers larger deviations from nominal values, it becomes more conservative, and average computing times tend to increase. This trend is evident in Table 4. When an uncertainty budget is fixed, increasing *Dev* generally leads to longer computing times across almost all uncertainty sets and budgets. The reason is analogous to the impact of budget size, since larger deviations require the algorithm to explore and generate more potential solutions that are later removed by the separation algorithm. From a decision-maker's perspective, investing in more accurate estimates of worst-case travel times, or in strategies to reduce deviations from nominal values, not only improves solution accuracy but can also decrease overall computational times.

Another interesting finding is that differences in computing performance in terms of time, optimality gaps and number of instances solved across uncertainty sets are relatively minor. Therefore the choice of uncertainty set should be primarily based on how well it represents the system under study rather than concerns about computational performance.

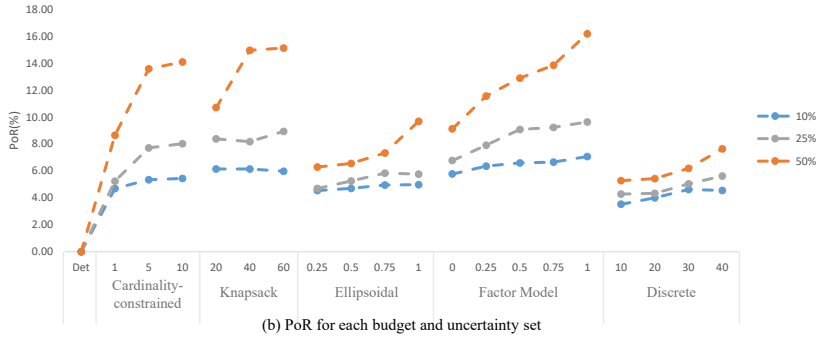
6.2 Robustness analysis

We analyze the impact of the proposed RO approach regarding the objective function value and robustness of the solutions. The robustness of a solution can be interpreted as the probability of constraint violation, which is an important metric for decision-making, as it provides a basis for assessing the trade-off between the solution's cost and its "safety" level. This information was estimated by implementing a Monte Carlo simulation with 10,000 scenarios in which the travel times follow a continuous uniform distribution from their nominal value to their maximum realization for a given deviation level Dev $\in \{10\%, 25\%, 50\%\}$.

Figure 1 depicts, for each combination of uncertainty set, budget, and maximum deviation, the average percentage of infeasible scenarios in the Monte Carlo simulation (Risk) and the average price-of-robustness (PoR), defined as the percentage cost increase of the robust solution over the deterministic one. The underlying values are reported in the tables of Appendix H. Across all uncertainty sets, increasing the budget of uncertainty improves the robustness of the solution, reducing the probability of constraint violation at the cost of higher expenses. This behavior is expected, since larger budgets correspond, by construction, to larger sets of realizations that the solution must remain feasible against; consequently, larger budgets yield safer but more constrained, and therefore more expensive, solutions. In some cases, however, the average cost associated with a larger budget is lower than that of a smaller one, e.g., for budgets $\Delta = 40$ and $\Delta = 60$ under Dev = 10%. This apparent inconsistency arises because some instances become infeasible under the



(a) Risk for each budget and uncertainty set



(b) PoR for each budget and uncertainty set

Figure 1: Price-of-robustness (PoR) and probability of constraint violation (Risk) for each uncertainty set.

larger budget and are consequently excluded from the average. This phenomenon can be observed in Table 5.

We note that each of the uncertainty sets behaved differently as the uncertainty budget increased. Some, like the ellipsoidal and factor model uncertainty sets, had solutions' risk level and costs changed more gradually while others, like the cardinality-constrained, had a sharp increase in the robustness of the solution between consecutive budgets. More specifically, the axis-parallel ellipsoidal uncertainty set never finds an empirically risk-free configuration for all instances and presents relatively small increases in solution costs between each budget, while the cardinality-constrained set quickly reaches a conservative solution configuration with near-zero risk in the simulated scenarios. We use the term *empirically risk-free* to denote solutions for which no constraint violation was observed across the 10,000 Monte Carlo scenarios. Although this suggests a very low infeasibility risk, out-of-sample scenarios may still lead to infeasible solutions.

Some uncertainty sets were able to provide empirically risk-free solutions, which allows them to be compared based solely on cost. For example, for instances with deviation of 50% over the nominal travel time, both the factor model (U_Ψ) and the knapsack (U_Δ) uncertainty sets provided empirically risk-free solutions for some budgets with different costs. The empirically risk-free solutions with set U_Ψ were, on average, 16.18% more expensive than the deterministic solution, while the same type of solutions cost between 14.95% and 15.12% more for the empirically risk-free solutions in set U_Δ with budgets $\Delta = 40$ and $\Delta = 60$. In these cases, the cheapest solution is the best choice because all three have the same risk of being infeasible.

Additionally, different configurations of the uncertainty set and budget can provide solutions that better satisfy the decision maker's strategy by offering a preferable trade-off between cost and risk, which would not be achievable with the cardinality-constrained set. In fact, the cardinality-constrained set failed to provide a wide range of solutions. Solutions with $\Gamma = 10$ showed similar or worse costs than those obtained with $\Gamma = 5$, while providing a marginal improvement in terms of safety level, indicating a limited usefulness of

these results.

Upon analyzing the behavior of the remaining uncertainty sets, we observed a more gradual transition under the factor model set (U_Ψ), offering interesting intermediate solutions with a competitive trade-off between cost and risk. However, in general, the empirically risk-free solutions obtained with this set were more expensive than those obtained through the other tested uncertainty sets, indicating a higher level of conservatism in the chosen budgets. Therefore, one should refrain from using this set unless it can accurately explain the logistical system (which is not the case since the studied instances are randomly generated and not bounded to any real-world factor).

The knapsack set (U_Δ) was relatively conservative in instances with small deviation, as even the budget $\Delta = 20$ was sufficient to absorb most of the deviation when considering deviations of 10% and 25% over the nominal travel time. Larger budgets, such as those of 40 and 60, resulted in risk-free solutions. Because the knapsack size is fixed, the decision maker must be especially careful when selecting a budget size that is compatible with the worst-case values of a given instance in this set; otherwise, the set may result in either fragile or overconservative solutions.

The axis-parallel ellipsoidal uncertainty set (U_Ω) also provided progressively more robust solutions. This set generally presented more intermediate options, but was unable to find empirically risk-free solutions with the chosen budgets. Nevertheless, solutions from this uncertainty set presented significant trade-offs that the decision maker could choose if willing to tolerate some risks. Notably, this set provided some solutions that were strictly superior to those of other uncertainty sets. For example, for instances with deviation of 50% over the nominal travel times with $\Omega = 0.75$, the solutions were, on average, less risky and less costly than those found with the cardinality-constrained set with a budget of $\Gamma = 1$. Similarly, in cases with a maximum deviation of 10% over the nominal travel time, solutions with $\Omega = 1$ were less risky and more economical than those obtained with the factor model set with $\Psi' = 0$. It is worth noting that, prior to this work, the only options available in the VRPTW literature were the cardinality-constrained uncertainty set (U_Γ), the knapsack uncertainty set (U_Δ) and the deterministic approach, and thus, one would lose these more competitive solutions.

The discrete uncertainty set (U_D) also produced distinct solutions that revealed significant trade-offs between cost and risk. As the number of scenarios increases, the solutions tend to become more conservative, since a larger number of potential realizations must be deemed feasible. Similar to the ellipsoidal uncertainty set, this approach could not generate empirically risk-free solutions, but it yielded solutions that performed better, on average, than those obtained with other sets and specific budgets. For example, in instances with a maximum deviation of 10% over the nominal value, considering 40 scenarios resulted in solutions with lower cost and risk than those obtained using the cardinality-constrained set with $\Gamma = 1$, the factor model with budgets $\Psi' = 0$ and 0.25, and the axis-parallel ellipsoidal uncertainty set with $\Omega = 1$. These results highlight the potential of the discrete uncertainty set when the scenarios are well constructed.

6.3 Design of an optimal robust route

In this section, we examine the structure of solutions generated with each uncertainty set. For this purpose, we analyze the optimal solutions of the (arbitrarily chosen) instance R112 with 25 customers and deviation of up to 50% over the nominal values under different uncertainty sets. In this instance, the customers were randomly placed around a depot in the center of the graph. The instance and its deterministic solution are illustrated in Figure 2, where the depot is represented by a square labeled 0, and the customers are represented by nodes labeled from 1 to 25. The four distinct routes in the solution are shown in different colors. Numerical details related to the travel times and time windows of the nodes on each route are provided in Table 10 in Appendix I.

The optimal deterministic solution for this particular instance has an objective value of 393 and a risk of constraint violation of 99.77%, indicating that it is highly vulnerable to uncertainty in travel times. Notably, the northwest (blue) and southwest (red) routes are particularly sensitive to variability. Specifically, the red route, the most fragile in the solution, becomes infeasible at node 6 if the total deviation exceeds 9, out of

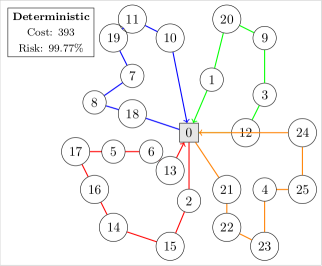


Figure 2: Optimal deterministic solution of instance R112 with an optimal value of 393 and a risk of 99.77%.

a maximum possible deviation of 44.4 (if all arcs up to node 6 reach their worst-case values). Similarly, the blue route fails to serve customer 10, the customer most likely not to be served on that route, if the total deviation exceeds 23.2 out of a maximum of 25.8 (computed by adding the deviation of every arc in the route up to node 10, excluding the deviation from arc (0,18), which is always absorbed by the opening time window of node 18). In contrast, the remaining routes (green and orange) are immunized to deviations; even if all arcs in their paths reach their worst-case values, the solution remains feasible. Therefore, when designing a robust solution, the emphasis is on modifying the blue and red routes, as will be analyzed next.

Figure 3 illustrates the optimal solutions for the cardinality-constrained uncertainty set, considering $\Gamma = 1$, 5, and 10. With $\Gamma = 1$, the solution remains identical to the deterministic one, as there is no arc in either the blue or red route whose maximum deviation exceeds the deviation buffers. Specifically, the maximum deviation for the red route is 9 and the maximum possible deviation for the blue route is 7.9. Neither of these is sufficient to exceed the buffers at nodes 6 and 10, respectively.

Conversely, with $\Gamma = 5$, the total deviation for both routes results in arrival times that exceed the closing time windows at these nodes. Consequently, in this more conservative scenario, the solution allocates node 6 to a new vehicle (purple route) and rearranges the order of the blue route so that node 10 is visited earlier. While the blue route of the deterministic solution is cheaper and has 9.8 fewer time units, the blue route of the more conservative solution reduces waiting time by 16.4 units, leading to an earlier expected arrival at node 10. This robust solution has a total cost of 423.8 units, which is 30.8 units more than the deterministic solution, but it reduces the risk of infeasibility, as estimated through Monte Carlo simulations, to 0%.

Although the solution with $\Gamma = 5$ did not fail in any of the 10,000 randomly generated scenarios, the southwest (red) route might still fail at customer 5 if all six arcs leading to that node reach their worst-case value. This is the main motivation for the changes observed in the solution with $\Gamma = 10$, which aims to serve node 5 earlier. However, the resulting solution is more expensive than the one obtained with $\Gamma = 5$, costing 429, even though it does not reduce the risk of constraint violations in the simulated scenarios. From a decision-maker's standpoint, there is little justification for adopting the solution with $\Gamma = 10$ over the solution with $\Gamma = 5$, as the former incurs higher costs without offering additional robustness. It is also worth noting that, since all routes in this solution visit fewer than ten arcs, it is fully protected against any uncertainty. This is effectively the same as if the instance had been optimized under the assumption that all parameters attain their worst-case values.

As seen in Figure 3, increasing the uncertainty budget in the cardinality-constrained set leads to more conservative solutions. This trend also applies to the other uncertainty sets. Figure 4 presents the solution for each of the remaining uncertainty sets that offers a significant trade-off between cost and constraint violation that could not be achieved using the cardinality-constrained set.

The knapsack uncertainty set produced two distinct solutions across the studied budgets: one for $\Delta = 20$, shown in Figure 4, and another for $\Delta > 20$, equivalent to the solution obtained with the cardinality-constrained uncertainty set with $\Gamma = 10$. The solution for $\Delta = 20$ has a cost of 409.9 and a risk of 7.30%, serving as an intermediate option between the deterministic solution and the empirically risk-free solution

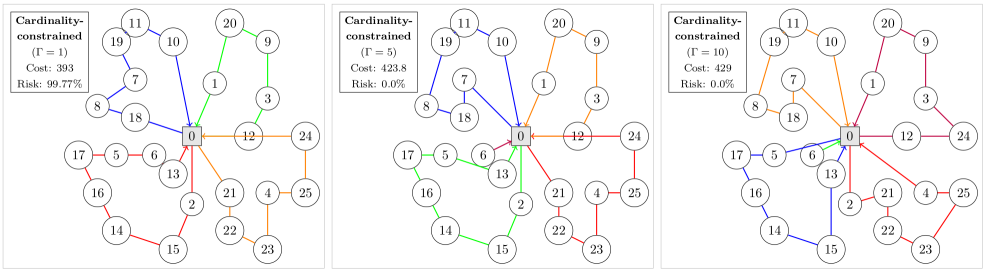


Figure 3: Optimal solution of instance R112 for the cardinality-constrained uncertainty set with $\Gamma = 1,\ 5,$ and 10.

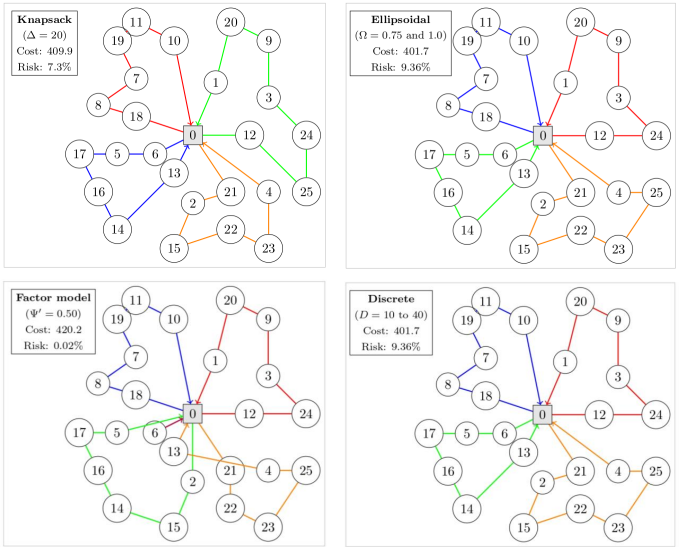


Figure 4: Examples of optimal solution of instance R112 for the knapsack, ellipsoidal, factor model, and discrete sets.

obtained with the cardinality-constrained set at $\Gamma = 5$. This solution reduces the number of nodes visited on the red route to allow node 6 to be visited in time with a maximum deviation of 20, since the vehicle would fail to serve customer 6 in the deterministic solution if the total deviation exceeds 9. However, the blue route remains unchanged because the buffer at node 10, the most critical node on that route, is larger than the budget of the knapsack containing all the arcs in the path (20). For larger values of Δ, this buffer is insufficient, and the algorithm returns a solution equivalent to the conservative one obtained with $\Gamma = 10$.

The solutions obtained with the axis-parallel ellipsoidal uncertainty set were identical to the deterministic one for $\Omega = 0.25$ and 0.5. The solution for $\Omega = 0.75$ and 1 is shown in Figure 4. This solution has a cost of 401.7 and a corresponding risk of 9.36%, representing another interesting trade-off between cost and risk. Notably, it is slightly cheaper and slightly riskier than the solution obtained with the knapsack uncertainty set and budget of $\Delta = 20$. Compared to the deterministic solution, the main difference is reassigning node 2 to the southeast (orange) route to allow node 6 to be visited earlier. When comparing this solution with the one obtained using the knapsack uncertainty set and $\Delta = 20$, we observe that the solution based on the ellipsoidal set is cheaper because it allows node 25 to remain on the orange route, whereas the deviation considered in the knapsack model prevents that assignment.

The factor model produced four unique solutions for this instance across the selected uncertainty budgets Ψ'. The first, with cost 401.7, was obtained for $\Psi' = 0$ and 0.25 and is identical to the one obtained with the ellipsoidal uncertainty set with $\Omega = 0.75$ and 1. The second solution, found for $\Psi' = 0.5$ and shown in Figure 4, has a cost of 420.2 and a risk of constraint violation of only 0.02%. This represents a compelling trade-off between cost and risk when compared to the empirically risk-free solution obtained with the cardinality-constrained set at $\Gamma = 5$, as it is 0.8% cheaper while maintaining a considerably low risk of constraint violation. From a decision-maker's standpoint, this solution is particularly attractive and should be carefully considered when selecting the final routing strategy. This solution resembles the one obtained with the cardinality-constrained set at $\Gamma = 5$, but differs in two main ways: it retains the northwest (blue) route of the deterministic solution, exploiting the large buffer at node 10, and it serves node 13 via the southeast (orange) route rather than the southwest (green) one, due to the larger travel-time deviation contributed by the southwest-quadrant factor. The final two solutions obtained with the factor model, corresponding to $\Psi' = 0.75$ and $\Psi' = 1$, have costs of 437.5 and 458.3, respectively, and no failed scenarios in the simulated experiments. This also indicates the importance of correctly parameterizing this uncertainty set, as it can potentially result in considerably conservative solutions, incurring extra costs with no practical benefit.

Finally, the discrete uncertainty set yielded the same solution for every budget D tested. This solution, shown in Figure 4, is identical to those obtained using the axis-parallel ellipsoidal uncertainty set with budgets $\Omega = 0.75$ and 1.

These results highlight the importance, from a decision-maker's perspective, of not relying on a single uncertainty set. Instead, considering a combination of different uncertainty sets enables the identification of a wider range of solutions with compelling trade-offs. For instance, if only the cardinality-constrained set were considered, as is common in literature, the decision-maker would be limited to choosing between the deterministic and the empirically risk-free solutions. Even when exploring additional values of Γ, the only alternative solution would be the one corresponding to budgets $\Gamma = 2$ to 4 [6], which coincides with the solution found by the discrete uncertainty set. Consequently, valuable solutions obtained through the knapsack and factor model uncertainty sets, each offering particularly interesting cost-risk trade-offs, would be overlooked.

7 Conclusion

We addressed the vehicle routing problem with time windows (VRPTW) under travel-time uncertainty considering the paradigm of RO. We proposed a polynomial time separation algorithm that is able to evaluate the robust feasibility of a solution for five different uncertainty sets, and that can be generalized for any closed, bounded and convex uncertainty set. This differs from most works in the literature, which concentrates only

on the traditional cardinality-constrained uncertainty set for the robust VRPTW (RVRPTW). To improve the computational performance of the separation algorithm, we derived closed-form expressions for each uncertainty set that compute the highest deviation in a subpath.

We evaluated the proposed algorithm within a branch-and-cut (BC) and a branch-price-and-cut (BPC) framework to solve RVRPTW instances under various uncertainty sets, including the cardinality-constrained, knapsack, ellipsoidal, factor model, and discrete sets. The BPC method incorporates a heuristic procedure which applies the separation algorithm to ensure route feasibility to generate an initial solution. The variety of solution approaches in this work also highlights the modularity and ease of implementation of the proposed separation algorithm.

We analyzed the impact of each uncertainty set on solution cost, robustness, and computational performance using benchmark instances from the literature, considering different uncertainty budgets and deviation levels. The results indicate that using different uncertainty sets can be advantageous in route generation. In particular, some of the studied uncertainty sets yielded solutions that achieved a trade-off between extra costs and the risk of constraint violation that was not achievable using the traditional cardinality-constrained uncertainty set. In fact, there were cases where using an alternative uncertainty set resulted in solutions that were strictly superior in terms of costs and risk to the ones obtained with the cardinality-constrained set. Furthermore, the computing times of the robust approach are comparable to those of the deterministic approach, with some configurations requiring less than 10% additional time.

For future work, the proposed separation algorithm can be easily implemented within dedicated metaheuristic methods, allowing larger instances to be solved. Another interesting direction for this research is the adaptation of this algorithm for other variants of the VRPTW, such as the pickup and delivery problem with time windows, the heterogeneous VRPTW and the split delivery VRPTW.

References

[1] Michel Gendreau, Ola Jabali, and Walter Rei. Future research directions in stochastic vehicle routing (50th anniversary invited article). *Transportation Science*, 50(4):1163–1173, 2016.

[2] Jorge Oyola, Halvard Arntzen, and David L. Woodruff. The stochastic vehicle routing problem, a literature review, part II: solution methods. *EURO Journal on Transportation and Logistics*, 6(4): 349–388, 2017. ISSN 2192-4376.

[3] Jorge Oyola, Halvard Arntzen, and David L. Woodruff. The stochastic vehicle routing problem, a literature review, part I: models. *EURO Journal on Transportation and Logistics*, 7(3):193–221, 2018. ISSN 2192-4376.

[4] Fernando Ordonez. Robust vehicle routing. In John J. Hasenbein, Paul Gray, and Harvey J. Greenberg, editors, *Risk and Optimization in an Uncertain World*, pages 153–178. INFORMS, Catonsville, MD. Chapter 7, 2010.

[5] Chrysanthos E. Gounaris, Wolfram Wiesemann, and Christodoulos A. Floudas. The robust capacitated vehicle routing problem under demand uncertainty. *Operations Research*, 61(3):677–693, 2013.

[6] Pedro Munari, Alfredo Moreno, Jonathan De La Vega, Douglas Alem, Jacek Gondzio, and Reinaldo Morabito. The robust vehicle routing problem with time windows: Compact formulation and branch-price-and-cut method. *Transportation Science*, 53(4):1043–1066, 2019.

[7] C. Lee, K. Lee, and S. Park. Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *Journal of the Operational Research Society*, 63:1294–1306, 2012.

[8] Agostinho Agra, Marielle Christiansen, Rosa Figueiredo, Lars Magnus Hvattum, Michael Poss, and Cristina Requejo. Layered formulation for the robust vehicle routing problem with time windows. In A. Ridha Mahjoub, Vangelis Markakis, Ioannis Milis, and Vangelis Th. Paschos, editors, *Combinatorial Optimization*, pages 249–260, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[9] Agostinho Agra, Marielle Christiansen, Rosa Figueiredo, Lars Magnus Hvattum, Michael Poss, and Cristina Requejo. The robust vehicle routing problem with time windows. *Computers & Operations Research*, 40(3):856–866, 2013.

[10] C. Hu, J. Lu, X. Liu, and G. Zhang. Robust vehicle routing problem with hard time windows under demand and travel time uncertainty. *Computers & Operations Research*, 94:139–153, 2018. ISSN 0305-0548.

[11] Akang Wang, Anirudh Subramanyam, and Chrysanthos Gounaris. Robust vehicle routing under uncertainty via branch-price-and-cut. *Optimization and Engineering*, 23:1895–1948, 2022.

[12] Artur Pessoa, Ruslan Sadykov, Eduardo Uchoa, and François Vanderbeck. A generic exact solver for vehicle routing and related problems. *Mathematical Programming*, 183(1-2):483–523, September 2020.

[13] Alex Abreu, Maria Battarra, Luciano Costa, and Pedro Munari. The robust pickup and delivery problem with time windows. *European Journal of Operational Research*, 2026. ISSN 0377-2217.

[14] Enrico Bartolini, Dominik Goeke, Michael Schneider, and Mengdie Ye. The robust traveling salesman problem with time windows under knapsack-constrained travel time uncertainty. *Transportation Science*, 55(2):371–394, 2021.

[15] Dimitris Bertsimas and Melvyn Sim. Robust discrete optimization and network flows. *Mathematical Programming*, 98(1):49–71, 2003.

[16] Rafael Campos, Leandro C. Coelho, and Pedro Munari. New formulations for the robust vehicle routing problem with time windows under demand and travel time uncertainty. *OR Spectrum*, 47(2):411–453, July 2024. ISSN 1436-6304.

[17] I. Sungur, F. Ordonez, and M. Dessouky. A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty. *IIE Transactions*, 40(5):509–523, 2008.

[18] Anirudh Subramanyam, Panagiotis P. Repoussis, and Chrysanthos E. Gounaris. Robust optimization of a broad class of heterogeneous vehicle routing problems under demand uncertainty. *INFORMS Journal on Computing*, 32(3):661–681, 2020.

[19] Bruno P. Bruck, Walton P. Coutinho, and Pedro Munari. The robust bike sharing rebalancing problem: Formulations and a branch-and-cut algorithm. *European Journal of Operational Research*, 325(1):67–80, 2025. ISSN 0377-2217.

[20] Chrysanthos E. Gounaris, Panagiotis P. Repoussis, Christos D. Tarantilis, Wolfram Wiesemann, and Christodoulos A. Floudas. An adaptive memory programming framework for the robust capacitated vehicle routing problem. *Transportation Science*, 50(4):1239–1260, 2016.

[21] Artur Alves Pessoa, Michael Poss, Ruslan Sadykov, and François Vanderbeck. Branch-cut-and-price for the robust capacitated vehicle routing problem with knapsack uncertainty. *Operations Research*, 69(3): 739–754, 2021.

[22] Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[23] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programming. *Operations Research Letters*, 25:1–13, 1999.

[24] Sebastián Ceria and Robert Stubbs. Incorporating estimation errors into portfolio selection: Robust portfolio construction. *Journal of Asset Management*, 7:109–127, 04 2006.

[25] Jens Lysgaard, Adam Letchford, and Richard Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100:423–445, 2004.

[26] Norbert Ascheuer, Matteo Fischetti, and Martin Grotschel. Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. *Mathematical Programming*, 90:475–506, 2001.

[27] Jens Lysgaard. Reachability cuts for the vehicle routing problem with time windows. *European Journal of Operational Research*, 175:210–223, 11 2006.

[28] Marius M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987.

Appendix A Proof of Theorem 3.1

Before proving the theorem, we establish an auxiliary result that expresses the earliest service start time induced by an arbitrary deviation vector in a form that isolates the role of the waiting times. This identity holds for any vector of arc deviations, independently of the uncertainty set, and is the main tool used in the proof that follows.

Lemma A.1. *Let $R = (r_0, \dots, r_{m+1})$ be a route with deterministic earliest service start times τ^0_r and waiting times σ_r defined by (12) for each node $r_i \in R$. For any realization η, with $\eta_{r_{i-1}r_i}$ representing the deviation on arc (r_{i-1}, r_i), and any node r_p, the earliest service start time at r_p induced by η satisfies*

$$\tau_{r_p}(\eta) = \tau^0_{r_p} + \max_{0\le k\le p} \Big[(w^a_{r_k} - \tau^0_{r_k}) + \sum_{i=k+1}^{p} \big(\eta_{r_{i-1}r_i} - \sigma_{r_i}\big)\Big], \tag{24}$$

where the index $k = p$ contributes the empty sum, equal to 0.

Proof. The induced service start times obey the recursion

$$\tau_{r_i}(\eta) = \max\big\{\, w^a_{r_i},\ \tau_{r_{i-1}}(\eta) + s_{r_{i-1}} + \bar{t}_{r_{i-1}r_i} + \eta_{r_{i-1}r_i}\big\}, \qquad \tau_{r_0}(\eta) = w^a_{r_0}. \tag{25}$$

Let $L_{r_i}(\eta) := \tau_{r_i}(\eta) - \tau^0_{r_i}$ be the schedule offset, given by the signed difference between the realized and the deterministic service start time at r_i. It is positive when the realization delays service relative to the deterministic schedule and negative when a reduction in upstream travel time allows service to start earlier. We show by induction that

$$L_{r_i}(\eta) = \max\big\{\, w^a_{r_i} - \tau^0_{r_i},\ L_{r_{i-1}}(\eta) + \eta_{r_{i-1}r_i} - \sigma_{r_i}\big\}, \qquad L_{r_0}(\eta) = 0. \tag{26}$$

The base case holds since $\tau_{r_0}(\eta) = w^a_{r_0} = \tau^0_{r_0}$. For the inductive step, the definition of σ_{r_i} in (12) gives $\tau^0_{r_{i-1}} + s_{r_{i-1}} + \bar{t}_{r_{i-1}r_i} = \tau^0_{r_i} - \sigma_{r_i}$. Rewriting the predecessor term through the definition of the offset as $\tau_{r_{i-1}}(\eta) = \tau^0_{r_{i-1}} + L_{r_{i-1}}(\eta)$ and substituting both into the recursion in (25) yields

$$\tau_{r_i}(\eta) = \max\big\{\, w^a_{r_i},\ \tau^0_{r_i} - \sigma_{r_i} + L_{r_{i-1}}(\eta) + \eta_{r_{i-1}r_i}\big\}. \tag{27}$$

Subtracting $\tau^0_{r_i}$ from both sides produces (26). The floor $w^a_{r_i} - \tau^0_{r_i} \le 0$ is the largest amount by which service at r_i can start ahead of its deterministic time. Because service cannot begin before the window opens, the realized start $\tau_{r_i}(\eta)$ is bounded below by $w^a_{r_i}$, so the offset $L_{r_i}(\eta)$ cannot fall below $w^a_{r_i} - \tau^0_{r_i}$. Reducing travel to node r_i beyond this point does not advance service starting time further, as the vehicle then waits for the window to open. The floor is zero at nodes where the deterministic start already coincides with the window opening ($\tau^0_{r_i} = w^a_{r_i}$) and strictly negative where the deterministic start is later ($\tau^0_{r_i} > w^a_{r_i}$); in the latter case a sufficiently negative deviation can make $L_{r_i}(\eta)$ negative, so that service at r_i begins ahead of the deterministic schedule.

Expanding recursion (26) from $L_{r_0} = 0$, each application either carries the accumulated offset forward, adding the increment $\eta_{r_{i-1}r_i} - \sigma_{r_i}$, or resets to the floor $w^a_{r_i} - \tau^0_{r_i} = 0$ at a node where the window binds. The surviving value at r_p is therefore the largest, over every node r_k taken as the last point at which the window is active, of the floor $w^a_{r_k} - \tau^0_{r_k}$ plus the deviations accumulated after r_k, $L_{r_p} = \max_{0\le k\le p}\Big[(w^a_{r_k} - \tau^0_{r_k}) + \sum_{i=k+1}^{p}\big(\eta_{r_{i-1}r_i} - \sigma_{r_i}\big)\Big]$, in which the index k identifies the last node where service starts at the time-window opening. Adding $\tau^0_{r_p}$ yields (24). □

With this identity in hand, we can now turn to the proof of the theorem.

Proof. Proof of Theorem 3.1. By the definition of σ_{r_i} in (12), the deterministic arrival can be computed with the following expression

$$\tau^0_{r_p} = \tau^0_{r_0} + \sum_{i=1}^{p} \big(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}} + \sigma_{r_i}\big), \tag{28}$$

so that the first two terms of (13) equal $\tau^0_{r_p}$. Combined with Lemma A.1, this shows that for any $s \in \{0, \dots, p-1\}$ the right-hand side of (13) is the arrival $\tau_{r_p}(\eta^{(s,p)})$ induced by the realization $\eta^{(s,p)}$, where $\eta^{(s,p)} \in U$ maximizes the subpath sum $\sum_{i=s+1}^{p} \eta_{r_{i-1}r_i}$; such a maximizer exists because the objective is linear and U is compact. As $\eta^{(s,p)}$ is feasible,

$$\tau^{r_s}_{r_p} = \tau_{r_p}(\eta^{(s,p)}) \le \max_{\eta \in U} \tau_{r_p}(\eta) \qquad \text{for every } s \in \{0, \dots, p-1\}, \tag{29}$$

and the right-hand maximum is attained, since U is compact and $\tau_{r_p}(\cdot)$ is continuous; denote it τ_{r_p} and let η^* be the realization that maximizes the arrival time at node r_p, τ_{r_p}.

It remains to show an index r_s attaining $\tau^{r_s}_{r_p} = \tau_{r_p}$. Applying Lemma A.1 to η^*, let k^* attain its inner maximum, so that

$$\tau_{r_p} = \tau_{r_p}(\eta^*) = \tau^0_{r_p} + (w^a_{r_{k^*}} - \tau^0_{r_{k^*}}) + \sum_{i=k^*+1}^{p} \big(\eta^*_{r_{i-1}r_i} - \sigma_{r_i}\big). \tag{30}$$

We may assume $k^* \le p-1$: if $k^* = p$, then by (30) the maximum in the invariant is attained by the empty suffix, so $\tau_{r_p} = \tau^0_{r_p} + (w^a_{r_p} - \tau^0_{r_p}) = w^a_{r_p}$, which equals the first argument $w^a_{r_p}$ of the outer maximization (14) and is therefore already attained there. For $k^* \le p-1$, we compare τ_{r_p} with the value produced at the index $s = k^*$. We do not claim that η^* maximizes any subpath sum; rather, the constructed realization $\eta^{(k^*,p)}$ maximizes the subpath sum over $(r_{k^*}, \dots, r_p)$, and thus

$$\sum_{i=k^*+1}^{p} \eta^{(k^*,p)}_{r_{i-1}r_i} \;\ge\; \sum_{i=k^*+1}^{p} \eta^*_{r_{i-1}r_i}. \tag{31}$$

Since the inner maximum in (13) at $s = k^*$ is taken over all $0 \le k \le p$, it is at least the value of the single candidate $k = k^*$, which carries the same floor $w^a_{r_{k^*}} - \tau^0_{r_{k^*}}$ as (30). This floor is identical on both sides of the comparison, so it does not affect it, and the bound reduces to the suffix deviation sum independently of whether $\eta^{(k^*,p)}$ induces waiting after r_{k^*}. Applying (31) and then (30),

$$\tau^{r_{k^*}}_{r_p} \;\ge\; \tau^0_{r_p} + (w^a_{r_{k^*}} - \tau^0_{r_{k^*}}) + \sum_{i=k^*+1}^{p} \big(\eta^{(k^*,p)}_{r_{i-1}r_i} - \sigma_{r_i}\big) \;\ge\; \tau^0_{r_p} + (w^a_{r_{k^*}} - \tau^0_{r_{k^*}}) + \sum_{i=k^*+1}^{p} \big(\eta^*_{r_{i-1}r_i} - \sigma_{r_i}\big) = \tau_{r_p}.$$

With the upper bound (29) at $s = k^*$, this forces $\tau^{r_{k^*}}_{r_p} = \tau_{r_p}$. Finally, since service cannot begin before the window opens, $\tau_{r_p} \ge w^a_{r_p}$; combining this with (29) and the attaining index k^* gives

$$\tau_{r_p} = \max\Big\{ w^a_{r_p},\ \max_{0 \le s \le p-1} \tau^{r_s}_{r_p} \Big\},$$

which is (14). The displayed form (13) follows by substituting (28) into the statement of Lemma A.1 for $\eta^{(s,p)}$.

□

Appendix B Proof of Proposition 1

Proof. Each realization of travel-time deviations is a vector $\eta \in U$ whose component η_{ij} is the effective deviation on arc $(i,j) \in \mathcal{A}$, and $\tau_{r_i}(\eta)$ denotes the earliest service start time at node r_i under realization η,

so that $\tau^*_{r_p} = \max_{\eta \in U} \tau_{r_p}(\eta)$. Since U is separable, each η_{ij} is given by its own dedicated variable ξ_{ij} through a map that vanishes at $\xi_{ij} = 0$, for instance, $\eta_{ij} = \xi_{ij}\hat{t}_{ij}$ for the ellipsoidal set, subject only to constraints of type $g(\xi) \le b$ nondecreasing in each $|\xi_{ij}|$ separately. Consequently, if $\eta \in U$ and η' is obtained from η by setting any subset of its components to zero, then $\eta' \in U$ as this relaxes every constraint $g(\xi) \le b$, so ξ remains feasible and $\eta' \in U$.

Adopt the convention $\tau^0_{r_0} = w^a_{r_0}$ at the depot, so that $0 \in \mathbb{L}$. By Lemma A.1, for every $\eta \in U$:

$$\tau_{r_p}(\eta) - \tau^0_{r_p} = \max_{0 \le k \le p} \Big[\big(w^a_{r_k} - \tau^0_{r_k} \big) + \sum_{j=k+1}^{p} \big(\eta_{r_{j-1}r_j} - \sigma_{r_j} \big) \Big].$$

Substituting $\sigma_{r_j} = \tau^0_{r_j} - \tau^0_{r_{j-1}} - s_{r_{j-1}} - \bar{t}_{r_{j-1}r_j}$ into the above expression and simplifying yields

$$\tau_{r_p}(\eta) = \max_{0 \le k \le p} \Big[w^a_{r_k} + \sum_{j=k+1}^{p} \big(\bar{t}_{r_{j-1}r_j} + s_{r_{j-1}} + \eta_{r_{j-1}r_j} \big) \Big]. \tag{32}$$

Maximizing over $\eta \in U$ and exchanging the two maximizations, which is valid because both are attained, gives

$$\tau^*_{r_p} = \max_{0 \le k \le p} \Big[w^a_{r_k} + \sum_{j=k+1}^{p} \big(\bar{t}_{r_{j-1}r_j} + s_{r_{j-1}} \big) + M_k \Big], \qquad M_k = \max_{\eta \in U} \sum_{j=k+1}^{p} \eta_{r_{j-1}r_j}, \tag{33}$$

where each M_k is attained because its objective is linear and U is compact. The term $k = p$ equals $w^a_{r_p}$, and for every $k \in \mathbb{L}$ with $k < p$ the k-th term coincides with $\tau^{r_k}_{r_p}$ as given by (15), since $w^a_{r_k} = \tau^0_{r_k}$. It therefore remains to show that every remaining index is dominated by an index of $\mathbb{L}$.

Two observations prepare the dominance argument. First, for every k, the maximum defining M_k is attained by a vector $\eta^{(k)}$ that is nonnegative and vanishes outside the arcs (r_{j-1}, r_j) with $k < j \le p$, which are pairwise distinct because the route is elementary, so each component of η lies unambiguously inside or outside this set. Starting from any maximizer, zeroing the components outside these arcs leaves the objective unchanged, zeroing the negative components among them cannot decrease it, and both operations preserve feasibility by the zero-substitution property established above. In particular, the realization optimizing the subpath $(r_k, \dots, r_p)$ can be taken to be zero on every arc preceding r_k, as claimed in the statement. Second, M is nonincreasing along the route: for $\ell \le k$, the vector $\eta^{(k)}$ is feasible for the problem defining M_ℓ, where its objective value is M_k because its components on the arcs between r_ℓ and r_k are zero; hence $M_\ell \ge M_k$.

Now let $k \notin \mathbb{L}$ with $k < p$, and let ℓ be the largest index of $\mathbb{L}$ smaller than k, which exists because $0 \in \mathbb{L}$. By the maximality of ℓ, no index j with $\ell < j \le k$ belongs to $\mathbb{L}$, and positive deterministic waiting at a node forces service to start at the opening of its time window, thus $\sigma_{r_j} = 0$ for all such j:

$$\tau^0_{r_k} = \tau^0_{r_\ell} + \sum_{j=\ell+1}^{k} \big(\bar{t}_{r_{j-1}r_j} + s_{r_{j-1}} \big) = w^a_{r_\ell} + \sum_{j=\ell+1}^{k} \big(\bar{t}_{r_{j-1}r_j} + s_{r_{j-1}} \big).$$

Splitting $\tau^{r_\ell}_{r_p}$ at index k and applying this identity,

$$\begin{aligned} \tau^{r_\ell}_{r_p} = w^a_{r_\ell} + \sum_{j=\ell+1}^{p} \big(\bar{t}_{r_{j-1}r_j} + s_{r_{j-1}} \big) + M_\ell &= \underbrace{w^a_{r_\ell} + \sum_{j=\ell+1}^{k} \big(\bar{t}_{r_{j-1}r_j} + s_{r_{j-1}} \big)}_{= \tau^0_{r_k}} + \sum_{j=k+1}^{p} \big(\bar{t}_{r_{j-1}r_j} + s_{r_{j-1}} \big) + M_\ell \\ &> w^a_{r_k} + \sum_{j=k+1}^{p} \big(\bar{t}_{r_{j-1}r_j} + s_{r_{j-1}} \big) + M_k, \end{aligned}$$

since $M_\ell \geq M_k$ and $\tau^0_{r_k} > w^a_{r_k}$. The k-th term of (33) is therefore strictly dominated by the ℓ-th term. Consequently,

$$\tau^*_{r_p} = \max\left\{w^a_{r_p},\ \max_{\ell\in\mathbb{L}} \tau^{r_\ell}_{r_p}\right\}, \tag{34}$$

with each $\tau^{r_\ell}_{r_p}$ given by (15), which establishes the restriction of the outer maximization (14) to $\mathbb{L}$. □

Appendix C Proof of the closed-form expressions

C.1 Cardinality-constrained uncertainty set

Theorem C.1. *The maximization subproblem associated with the cardinality-constrained uncertainty set in expression* (15) *admits the closed-form solution*

$$\max_{\xi\in[0,1]^{Ps}}\left\{\sum_{i=s+1}^{p} \xi_{r_{i-1}r_i}\,\hat{t}_{r_{i-1}r_i}\ :\ \sum_{i=s+1}^{p}\xi_{r_{i-1}r_i}\leq\Gamma\right\} = \sum_{i=1}^{\min\{Ps,\,\lfloor\Gamma\rfloor\}}\hat{t}_{(i)} + \mathbf{1}_{\{\Gamma<Ps\}}\,(\Gamma-\lfloor\Gamma\rfloor)\,\hat{t}_{(\lfloor\Gamma\rfloor+1)}, \tag{35}$$

where $\hat{t}_{(i)}$ denotes the deviations along the subpath, indexed in non-increasing order, $Ps = p-s$ is the number of the arcs on subpath $(r_s,\dots,r_p)$, and $\mathbf{1}_{\{\cdot\}}$ is the indicator function.

Proof. The subproblem in (35) is an instance of the continuous (fractional) knapsack problem with Ps items, each of unit weight and value $\hat{t}_{r_{i-1}r_i}$, $i\in\{s+1,\dots,p\}$, and total capacity Γ. Since all weights are equal, the value-to-weight ratio reduces to the value itself, and the standard greedy algorithm constructs an optimal solution by selecting items in non-increasing order of $\hat{t}_{r_{i-1}r_i}$.

Formally, the greedy procedure sets

$$\xi^*_{(i)} = \begin{cases} 1, & i = 1,\dots,\min\{Ps,\lfloor\Gamma\rfloor\},\\ \Gamma-\lfloor\Gamma\rfloor, & i=\lfloor\Gamma\rfloor+1,\ \text{ if } \Gamma<Ps,\\ 0, & \text{otherwise.}\end{cases}$$

Substituting ξ^* into the objective function yields the right-hand side of (35). When $\Gamma\geq Ps$, the budget is large enough to fully exploit every arc, and the indicator term vanishes. □

C.2 Knapsack uncertainty set

Theorem C.2. *Assume that the knapsack subsets $\{S_l\}_{l\in L}$ are pairwise disjoint and cover the arc set. Then the maximization subproblem associated with the knapsack uncertainty in expression* (15) *set admits the closed-form solution*

$$\max_{\xi\in[0,1]^{Ps}}\left\{\sum_{i=s+1}^{p} \xi_{r_{i-1}r_i}\,\hat{t}_{r_{i-1}r_i}\ :\ \sum_{\substack{i=s+1\\(r_{i-1},r_i)\in S_l}}^{p}\xi_{r_{i-1}r_i}\,\hat{t}_{r_{i-1}r_i}\leq\Delta_l,\ l\in L\right\} = \sum_{l\in L}\min\left\{\Delta_l,\ \sum_{\substack{i=s+1\\(r_{i-1},r_i)\in S_l}}^{p}\hat{t}_{r_{i-1}r_i}\right\}. \tag{36}$$

Proof. Introduce the change of variable $v_{r_{i-1}r_i} = \xi_{r_{i-1}r_i}\,\hat{t}_{r_{i-1}r_i}$, which represents the absolute deviation actually realized on arc (r_{i-1},r_i). The subproblem (36) can then be rewritten as the linear program

$$\max_{v\in\mathbb{R}^{Ps}_+}\left\{\sum_{i=s+1}^{p} v_{r_{i-1}r_i}\ :\ v_{r_{i-1}r_i}\leq\hat{t}_{r_{i-1}r_i},\ \sum_{\substack{i=s+1\\(r_{i-1},r_i)\in S_l}}^{p} v_{r_{i-1}r_i}\leq\Delta_l,\ l\in L\right\}. \tag{37}$$

Associate dual variables $\pi_{r_{i-1}r_i} \geq 0$ with the upper bound constraints and $\theta_l \geq 0$ with the knapsack constraints. By linear programming duality, (37) is equivalent to

$$\min_{\substack{\pi \in \mathbb{R}_+^{Ps} \\ \theta \in \mathbb{R}_+^{|L|}}} \left\{ \sum_{i=s+1}^{p} \pi_{r_{i-1}r_i} \hat{t}_{r_{i-1}r_i} + \sum_{l \in L} \Delta_l \, \theta_l \;:\; \pi_{r_{i-1}r_i} + \sum_{\substack{l \in L \\ (r_{i-1}, r_i) \in S_l}} \theta_l \geq 1,\ i = s+1, \ldots, p \right\}. \tag{38}$$

Since the knapsacks are disjoint, each arc (r_{i-1}, r_i) belongs to exactly one subset; let $l(i) \in L$ denote the index of this knapsack. Then $\sum_{l \in L,\, (r_{i-1}, r_i) \in S_l} \theta_l = \theta_{l(i)}$, so the dual constraint simplifies to $\pi_{r_{i-1}r_i} + \theta_{l(i)} \geq 1$.

We can now construct an explicit dual-feasible solution that attains the claimed objective value. For each $l \in L$, define

$$\theta_l^* = \begin{cases} 1, & \text{if } \Delta_l \leq \sum_{\substack{i=s+1 \\ (r_{i-1}, r_i) \in S_l}}^{p} \hat{t}_{r_{i-1}r_i}, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$\pi_{r_{i-1}r_i}^* = 1 - \theta_{l(i)}^* \in \{0, 1\}.$$

By construction, (π^*, θ^*) is dual feasible and non-negative. Substituting into the dual objective function yields

$$\sum_{l \in L} \left[\theta_l^* \, \Delta_l + \sum_{\substack{i=s+1 \\ (r_{i-1}, r_i) \in S_l}}^{p} (1 - \theta_l^*) \, \hat{t}_{r_{i-1}r_i} \right] = \sum_{l \in L} \min \left\{ \Delta_l, \sum_{\substack{i=s+1 \\ (r_{i-1}, r_i) \in S_l}}^{p} \hat{t}_{r_{i-1}r_i} \right\}.$$

To see that (π^*, θ^*) is optimal, observe that any dual-feasible solution must satisfy $\pi_{r_{i-1}r_i} + \theta_{l(i)} \geq 1$; reducing either variable below the value prescribed above would violate this constraint, while increasing either variable would increase the objective because all coefficients are non-negative. Hence (π^*, θ^*) is optimal and, by strong LP duality, the primal optimum coincides with the right-hand side of (36).

□

C.3 Ellipsoidal uncertainty set

Theorem C.3. *The maximization subproblem associated with the axis-parallel ellipsoidal uncertainty in expression* (15) *set admits the closed-form solution:*

$$\max_{\xi_{ij} \in \mathbb{R}} \Big(\sum_{i=s+1}^{p} \xi_{r_{i-1}r_i} \hat{t}_{r_{i-1}r_i} : \sum_{i=s+1}^{p} (\xi_{r_{i-1}r_i})^2 \leq \Omega^2 \Big) = \Omega \sqrt{\sum_{i=s+1}^{p} (\hat{t}_{r_{i-1}r_i})^2}, \tag{39}$$

for $\Omega \in [0, 1]$.

Proof. Treating $\xi = (\xi_{r_{i-1}r_i})$ and $\hat{t} = (\hat{t}_{r_{i-1}r_i})$ as vectors in $\mathbb{R}^{Ps}$, the Cauchy–Schwarz inequality gives

$$\sum_{i=s+1}^{p} \xi_{r_{i-1}r_i} \, \hat{t}_{r_{i-1}r_i} \;\leq\; \sqrt{\sum_{i=s+1}^{p} \xi_{r_{i-1}r_i}^2} \sqrt{\sum_{i=s+1}^{p} \hat{t}_{r_{i-1}r_i}^2}$$

based on the feasibility condition of the uncertainty set $\sum_{i=s+1}^{p} (\xi_{r_{i-1}r_i})^2 \leq \Omega^2$, the following inequality holds:

$$\sqrt{\sum_{i=s+1}^{p} \xi_{r_{i-1}r_i}^2} \sqrt{\sum_{i=s+1}^{p} \hat{t}_{r_{i-1}r_i}^2} \;\leq\; \Omega \sqrt{\sum_{i=s+1}^{p} \hat{t}_{r_{i-1}r_i}^2},$$

Equality in Cauchy–Schwarz holds if and only if ξ is a non-negative scalar multiple of $\hat{t}$. Setting

$$\xi^*_{r_{i-1}r_i} = \frac{\Omega\, \hat{t}_{r_{i-1}r_i}}{\sqrt{\sum_{k=s+1}^{p} \hat{t}^2_{r_{k-1}r_k}}}, \qquad i = s+1, \dots, p, \tag{40}$$

yields

$$\sum_{i=s+1}^{p} (\xi^*_{r_{i-1}r_i})^2 = \frac{\Omega^2 \sum_{i=s+1}^{p} \hat{t}^2_{r_{i-1}r_i}}{\sum_{k=s+1}^{p} \hat{t}^2_{r_{k-1}r_k}} = \Omega^2,$$

so the ellipsoidal constraint is active at ξ^*, and the upper bound is attained. This establishes (39). □

C.4 Factor model uncertainty set

For notational convenience, throughout this proof we write

$$\Psi_f \;:=\; \sum_{i=s+1}^{p} \Psi_{r_{i-1}r_i f}, \qquad f = 1, \dots, F,$$

for the aggregate factor loading along the subpath, and assume that the columns of the loading matrix Ψ are ordered so that $\Psi_1 \geq \Psi_2 \geq \cdots \geq \Psi_F$. Let also

$$f^+ \;=\; \left\lceil \frac{(1+\beta)F}{2} \right\rceil, \qquad f^- \;=\; \max\left\{ \left\lceil \frac{(1-\beta)F}{2} \right\rceil, 1 \right\}.$$

Theorem C.4. *The maximization subproblem associated with the factor model uncertainty set admits the closed-form solution*

$$\max_{\xi \in [-1,1]^F} \left\{ \sum_{f=1}^{F} \xi_f\, \Psi_f \;:\; -\beta F \leq \sum_{f=1}^{F} \xi_f \leq \beta F \right\} \;=\; \min_{\lambda \in \Lambda} \left\{ \sum_{f=1}^{F} \big|\Psi_f - \lambda\big| + \beta F\, |\lambda| \right\}, \tag{41}$$

where $\Lambda = \{0,\ \Psi_{f^+},\ \Psi_{f^-}\}$.

Proof. This proof is based on the one described in [5]. The primal problem is a linear program in ξ. Associating dual multipliers $\eta^+_f, \eta^-_f \geq 0$ with the upper and lower bounds on ξ_f, and $\theta^+, \theta^- \geq 0$ with the upper and lower bounds on $\sum_f \xi_f$, LP duality yields the equivalent problem

$$\min_{\substack{\eta^+, \eta^- \in \mathbb{R}^F_+ \\ \theta^+, \theta^- \in \mathbb{R}_+}} \left\{ \sum_{f=1}^{F} (\eta^+_f + \eta^-_f) + \beta F\, (\theta^+ + \theta^-) \;:\; (\eta^+_f - \eta^-_f) + (\theta^+ - \theta^-) = \Psi_f,\ f = 1, \dots, F \right\}. \tag{42}$$

Since all objective coefficients in (42) are non-negative, there exists an optimal solution in which $\eta^+_f \eta^-_f = 0$ for every f and $\theta^+\theta^- = 0$. If both components of a pair were strictly positive, reducing them by the smaller of the two values would preserve dual feasibility while strictly decreasing the objective. Defining the unrestricted variables $\eta_f := \eta^+_f - \eta^-_f$ and $\theta := \theta^+ - \theta^-$, the relations $\eta^+_f + \eta^-_f = |\eta_f|$ and $\theta^+ + \theta^- = |\theta|$ hold at any such optimal solution, so (42) reduces to

$$\min_{\substack{\eta \in \mathbb{R}^F \\ \theta \in \mathbb{R}}} \left\{ \sum_{f=1}^{F} |\eta_f| + \beta F\, |\theta| \;:\; \eta_f + \theta = \Psi_f,\ f = 1, \dots, F \right\}. \tag{43}$$

The equality constraint in (43) gives $\eta_f = \Psi_f - \theta$, and substituting back produces the univariate convex program

$$\min_{\theta \in \mathbb{R}} \ \Phi(\theta) \ := \ \sum_{f=1}^{F} \big|\Psi_f - \theta\big| + \beta F \, |\theta|. \tag{44}$$

The function Φ is convex and piecewise linear, with breakpoints at $\theta = 0$ and at $\theta = \Psi_f$ for $f = 1, \dots, F$. Since $\Phi(\theta) \to +\infty$ as $|\theta| \to \infty$, the minimum is attained at one of these breakpoints.

To identify the optimal breakpoint, suppose first that $\theta^* > 0$. The left and right derivatives of Φ at a generic point θ are

$$\Phi'_-(\theta) = \big|\{f : \Psi_f < \theta\}\big| - \big|\{f : \Psi_f \geq \theta\}\big| + \beta F,$$
$$\Phi'_+(\theta) = \big|\{f : \Psi_f \leq \theta\}\big| - \big|\{f : \Psi_f > \theta\}\big| + \beta F.$$

Let $\ell \in \{1, \dots, F\}$ index the candidate breakpoint, so that $\theta^* = \Psi_\ell$ corresponds to choosing the ℓ-th largest aggregate loading. Under the ordering $\Psi_1 \geq \cdots \geq \Psi_F$, evaluating these derivatives at $\theta^* = \Psi_\ell$ gives $\Phi'_-(\Psi_\ell) = (1+\beta)F - 2\ell$ and $\Phi'_+(\Psi_\ell) = (1+\beta)F - 2\ell + 2$. By convexity of Φ, the point $\theta^* = \Psi_\ell$ is optimal if and only if $\Phi'_-(\theta^*) \leq 0 \leq \Phi'_+(\theta^*)$, which simplifies to

$$\frac{(1+\beta)F}{2} \ \leq \ \ell \ \leq \ \frac{(1+\beta)F}{2} + 1.$$

The smallest integer satisfying these inequalities is $f^+ = \lceil (1+\beta)F/2 \rceil$, so $\theta^* = \Psi_{f^+}$. When ties are present in the Ψ_f, the same argument applies with the subdifferential replacing left and right derivatives, and the optimum is attained at the same breakpoint value. By a symmetric argument, if $\theta^* < 0$, the optimum is attained at $\theta^* = \Psi_{f^-}$; the outer max in the definition of f^- ensures the index is well defined when $(1-\beta)F/2 = 0$. The remaining candidate $\theta^* = 0$ corresponds to the case in which the constraint on $\sum_f \xi_f$ is non-binding.

Hence the optimum of Φ is attained for some $\lambda \in \Lambda = \{0, \Psi_{f^+}, \Psi_{f^-}\}$, and substituting $\theta = \lambda$ in (44) yields the right-hand side of (41).

□

Corollary C.4.1. *Let $\lambda^* \in \Lambda$ attain the minimum in* (41)*, and set $a = |\{f : \Psi_f > \lambda^*\}|$, $b = |\{f : \Psi_f < \lambda^*\}|$, and $T = \{f : \Psi_f = \lambda^*\}$. A worst-case realization ξ^* attaining the left-hand side of* (41) *is recovered in $O(F)$ operations by setting $\xi^*_f = \operatorname{sign}(\Psi_f - \lambda^*)$ for $f \notin T$ and, when $T \neq \emptyset$, $\xi^*_f = S/|T|$ for $f \in T$, where $S = \operatorname{sign}(\lambda^*)\beta F - (a-b)$ if $\lambda^* \neq 0$, and S is the projection of 0 onto the interval $\big[\max\{-|T|, -\beta F - (a-b)\}, \min\{|T|, \beta F - (a-b)\}\big]$ if $\lambda^* = 0$. The induced per-arc deviations $\delta^*_{r_{i-1}r_i} = \sum_{f=1}^{F} \xi^*_f \Psi_{r_{i-1}r_i f}$ satisfy $\sum_{i=s+1}^{p} \delta^*_{r_{i-1}r_i} = \sum_{f=1}^{F} \xi^*_f \Psi_f$, the optimal value of* (41)*.*

Proof. Fix the subpath $(r_s, \dots, r_p)$ and write $\Psi_f := \Psi_f^{(s,p)} = \sum_{i=s+1}^{p} \Psi_{r_{i-1}r_i f}$, ordered so that $\Psi_1 \geq \cdots \geq \Psi_F$. By LP duality applied to (41), the optimal multiplier of the budget constraint equals the minimizing breakpoint: writing it as $\theta = \theta^+ - \theta^-$, with $\theta^+, \theta^- \geq 0$ the multipliers associated with $\sum_f \xi_f \leq \beta F$ and $-\sum_f \xi_f \leq \beta F$, we have $\theta = \lambda^*$. Fixing $\theta = \lambda^*$, the Lagrangian of the inner maximization separates across factors into the scalar problems $\max_{-1 \leq \xi_f \leq 1} (\Psi_f - \lambda^*)\xi_f$, so $\xi^*_f = \operatorname{sign}(\Psi_f - \lambda^*)$ is optimal for every $f \notin T$, while any value in $[-1, 1]$ is optimal for a tied factor (those in subset T), provided the budget constraint remains satisfied and, when $\lambda^* \neq 0$, tight on the appropriate side.

If $\lambda^* \neq 0$, then $\theta^+ > 0$ or $\theta^- > 0$, and by complementary slackness $\sum_f \xi^*_f = \operatorname{sign}(\lambda^*)\beta F$. The non-tied factors contribute $a - b$ to this sum, so the tied ones must supply $S = \operatorname{sign}(\lambda^*)\beta F - (a-b)$. Evaluating the one-sided derivatives of the dual objective Φ at λ^* (stated for $\lambda^* > 0$; the case $\lambda^* < 0$ is symmetric) gives $\Phi'_-(\lambda^*) = b - (a + |T|) + \beta F$ and $\Phi'_+(\lambda^*) = (b + |T|) - a + \beta F$, so the breakpoint optimality condition $\Phi'_-(\lambda^*) \leq 0 \leq \Phi'_+(\lambda^*)$ is equivalent to $-|T| \leq S \leq |T|$, and the equal split $\xi^*_f = S/|T|$ is feasible. If instead $\lambda^* = 0$, the

budget constraint need not be active: the tied factors are only required to keep $\sum_f \xi_f^* \in [-\beta F, \beta F]$, that is, to supply a sum in $\big[\max\{-|T|, -\beta F - (a-b)\},\ \min\{|T|, \beta F - (a-b)\}\big]$. This interval is nonempty, because the breakpoint optimality conditions $\Phi'_-(0) = b - (a + |T|) - \beta F \le 0$ and $\Phi'_+(0) = (b + |T|) - a + \beta F \ge 0$ amount to $|a-b| \le \beta F + |T|$, and it contains S by construction. In both cases every tied entry lies in $[-1, 1]$, since $|S| \le |T|$.

Optimality of ξ^* follows by direct substitution: using the identities $\operatorname{sign}(\Psi_f - \lambda^*)\Psi_f = |\Psi_f - \lambda^*| + \operatorname{sign}(\Psi_f - \lambda^*)\lambda^*$ and $\lambda^*(a - b + S) = \beta F|\lambda^*|$, valid in both cases,

$$\sum_{f=1}^{F} \xi_f^* \Psi_f = \sum_{f \notin T} \operatorname{sign}(\Psi_f - \lambda^*)\Psi_f + \lambda^* S = \sum_{f=1}^{F} \big|\Psi_f - \lambda^*\big| + \beta F \big|\lambda^*\big| = \Phi(\lambda^*),$$

the optimal value of (41). Finally, $\sum_{i=s+1}^{p} \delta^*_{r_{i-1}r_i} = \sum_{f=1}^{F} \xi_f^* \sum_{i=s+1}^{p} \Psi_{r_{i-1}r_i f} = \sum_{f=1}^{F} \xi_f^* \Psi_f$, by exchanging the order of summation. Substituting the per-arc deviations δ^* of Corollary C.4.1 for $\eta^{(s,p)}$ in (13) gives the factor-model closed form

$$\tau^{r_s}_{r_p} = \tau^0_{r_s} + \sum_{i=s+1}^{p} \big(\bar{t}_{r_{i-1}r_i} + s_{r_{i-1}} + \sigma_{r_i}\big) + \max_{0 \le k \le p} \Big[(w^a_{r_k} - \tau^0_{r_k}) + \sum_{i=k+1}^{p} \big(\delta^{(s,p)}_{r_{i-1}r_i} - \sigma_{r_i}\big) \Big], \tag{45}$$

with $\delta^{(s,p)}_{r_{i-1}r_i} = \sum_{f=1}^{F} \xi_f^* \Psi_{r_{i-1}r_i f}$.

□

Remark. *When $T \neq \emptyset$, the maximizer of* (41) *is not unique: any feasible allocation of S among the tied factors is optimal, and different allocations induce different per-arc deviations, hence different intermediate values $\tau^{r_s}_{r_p}$ in* (13)*, since a tied factor's loading over a range $(r_k, \ldots, r_p)$ with $k \neq s$ need not equal its aggregate $\Psi_f = \lambda^*$ over the subpath. Nevertheless, by Theorem 3.1 and the remark following it, the outer maximization* (14) *returns the same worst-case arrival τ_{r_p} for every choice of maximizer, so the fixed split of Corollary C.4.1 may be used throughout. Reoptimizing the tied allocation for each reset point k is a valid alternative that can only enlarge the intermediate values $\tau^{r_s}_{r_p}$, possibly triggering the infeasibility test at an earlier predecessor, but it leaves τ_{r_p} unchanged.*

C.5 Discrete uncertainty set

Theorem C.5. *The maximization subproblem associated with the discrete uncertainty set in expression* (13) *admits the closed-form solution*

$$\max_{t \in U^D} \sum_{i=s+1}^{p} \big(t_{r_{i-1}r_i} - \bar{t}_{r_{i-1}r_i}\big) = \max\left\{ \sum_{i=s+1}^{p} \hat{t}^{(j)}_{r_{i-1}r_i} \ :\ j \in \{1, \ldots, D\} \right\}. \tag{46}$$

Proof. The objective is a linear function of the realization t, and the uncertainty set $U^D = \operatorname{conv}\{t^{(1)}, \ldots, t^{(D)}\}$ is a bounded polytope whose vertices are (a subset of) the scenarios $t^{(j)}$. By the fundamental theorem of linear programming, the maximum of a linear function over a bounded polytope is attained at a vertex. Therefore, the optimum is attained at some scenario $t^{(j^*)}$, and the optimal value equals $\max_{j \in \{1,\ldots,D\}} \sum_{i=s+1}^{p} \hat{t}^{(j)}_{r_{i-1}r_i}$.

We note that scenarios dominated componentwise by another scenario can be omitted without affecting the optimum, since they cannot be optimal vertices of the maximization. □

Appendix D Algorithm example

This appendix illustrates the behavior of the proposed feasibility algorithm through an example under the axis-parallel ellipsoidal uncertainty set. The example highlights a key distinction between travel-time and

demand uncertainty. In robust optimization problems under demand uncertainty, the worst-case realization is typically obtained by maximizing the cumulative deviation along the route. Under travel-time uncertainty with (non-zero) opening time windows, however, this approach is no longer valid, as part of the realized delay may be absorbed by waiting times. Consequently, a realization with a smaller cumulative travel-time deviation may produce a later arrival than another realization with a larger total delay.

Figure 5 illustrates this phenomenon by computing the latest arrival time at node r_5 in the route $R = (r_0, \ldots, r_5, \ldots, r_{m+1})$. Each row assumes that travel-time deviations are considered only on the route suffix beginning after node r_s, where $s \in \{0, \ldots, 4\}$. The node time windows are shown above each customer, while the nominal travel times $\bar{t}_{ij}$ and the maximum deviations $\hat{t}_{ij}$ are displayed below and above the corresponding arcs, respectively. Service times are omitted for simplicity.

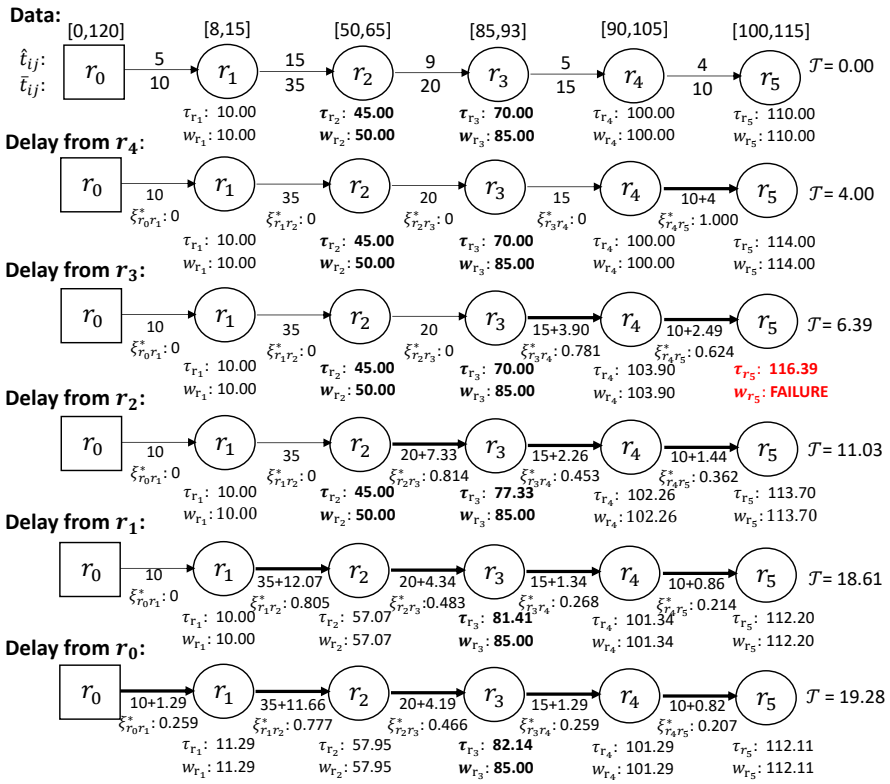


Figure 5: Illustration of the feasibility algorithm.

For each route suffix, the worst-case realization is computed under an axis-parallel ellipsoidal uncertainty set with $\Omega = 1$ by solving

$$\max_{\xi \in \mathbb{R}^{n \times n}} \left\{ \sum_{i=s+1}^{5} \xi_{r_{i-1}r_i} \hat{t}_{r_{i-1}r_i} \; : \; \sum_{i=s+1}^{5} (\xi_{r_{i-1}r_i})^2 \leq 1 \right\}, \tag{47}$$

Bold arcs indicate where deviations are allowed. For each case, the figure reports the optimal realization, the corresponding arrival time τ_i, the service start time at each customer, and the cumulative travel-time deviation

$$\mathcal{T} = \sum_{i=s+1}^{5} \xi^*_{r_{i-1}r_i} \hat{t}_{r_{i-1}r_i}. \tag{48}$$

The worst-case arrival time at node r_5 is obtained when deviations are considered only on the suffix beginning after node r_3, yielding $\tau_{r_5} = 116.39$ with a cumulative deviation of $\mathcal{T} = 6.39$. Interestingly, allowing deviations on additional arcs produces larger cumulative deviations but earlier arrivals at node r_5. This occurs because the waiting time induced by the opening time window at node r_3 completely absorbs the delays accumulated before this node. Hence, unlike robust optimization problems under demand uncertainty, maximizing the cumulative travel-time deviation does not necessarily identify the realization yielding the latest arrival time. This observation motivates the proposed feasibility algorithm, which explicitly accounts for the interaction between travel-time deviations and waiting times.

Appendix E Robust graph preprocessing

This appendix details the robust preprocessing procedure summarized in Section 4.1, which extends the deterministic preprocessing rules of Ascheuer et al. [26] to account for travel-time uncertainty, following Bartolini et al. [14].

The total time, including travel and service times, of any subpath P under uncertainty set U is denoted S_P^U. Parameter α_{ij} represents the maximum deviation on arc (i,j) for a given uncertainty set U, computed using the expressions in Table 6; these closed-form expressions are obtained from the non-deterministic components of the formulations in Table 1 by specializing them to a path $R = (i,j)$, with $r_s = i$ and $r_p = j$.

Table 6: Mathematical expression of parameter α_{ij} used in the time windows tightening and basic robust arc removal steps.

Uncertainty set	**Closed-Form**
Cardinality-constrained (U_Γ)	$\alpha_{ij} = \min\{\hat{t}_{ij}, \Gamma\hat{t}_{ij}\}$
Knapsack (U_Δ)	$\alpha_{ij} = \min\{\hat{t}_{ij}, \sum_{\substack{l\in L \\ (i,j)\in S_l}} \Delta_l\}$
Ellipsoidal (U_Ω)	$\alpha_{ij} = \min\{\hat{t}_{ij}, \Omega\hat{t}_{ij}\}$,
Factor model (U_Ψ)	$\alpha_{ij} = \min\Big(\sum_{f=1}^{F} \lvert\lambda - \Psi_{ijf}\rvert + \beta F \lvert\lambda\rvert : \lambda \in \Big\{0, \Psi_{ijf+}, \Psi_{ijf-}\Big\}\Big)$
Discrete (U_D)	$\alpha_{ij} = \max_{d\in\{1\ldots,D\}}\{\hat{t}_{ij}^d\}$

With these definitions in place, we detail the arc-removal and time-window-tightening procedures below. These steps are applied iteratively until no further changes occur in the instance graph $\mathcal{G}$ at the end of an iteration, defined as a complete execution of all the proposed steps.

Time windows tightening

Step 1: Let $\bar{w}_i^a := \min\{w_i^b, \min_{\substack{(j,i)\in\mathcal{A} \\ j\neq n+1}} \{w_j^a + s_j + \bar{t}_{ji}\}\}$ and set $w_i^a := \max\{\bar{w}_i^a, w_i^a\}, \forall i \in \mathcal{N}^*$;

Step 2: Let $\bar{w}_i^b := \max\{w_i^a, \bar{t}_{0i} + \alpha_{0i}, \max_{\substack{(i,j)\in\mathcal{A} \\ j\neq 0}} \{w_j^b - s_i - \bar{t}_{ij} - \alpha_{ij}\}\}$ and set $w_i^b := \min\{\bar{w}_i^b, w_i^b\}, \forall i \in \mathcal{N}$;

Basic robust arc removal

Step 1: Remove from $\mathcal{A}$ all arcs (i,j) in which $w_i^a + s_i + \bar{t}_{ij} + \alpha_{ij} > w_j^b$;

Generation of precedence list for arc removal

Step 1: For each pair of nodes i and j, flag node i that precedes j ($i \prec j$) if, and only if, $w_j^a + S_{(j,i)}^U > w_i^b$;

Step 2: For each pair of nodes i and j, if there is a node $k \in \mathcal{N}$ such that $i \prec k$ and $k \prec j$, then $i \prec j$. Repeat this step until no new precedence is detected.

Arc removal due precedence

Step 1: Remove from $\mathcal{A}$ all arcs (j,i) in which node i precedes node j ($i \prec j$).

Step 2: Remove all arcs (i,j) such that:

- $k \prec i$ or $k \prec j$ or $w_i^a + S^U_{(i,j)\cup(j,k)} > w_k^b$,
  and
- $i \prec k$ or $j \prec k$ or $w_k^a + S^U_{(k,i)\cup(i,j)} > w_j^b$,
  for some $k \in \mathcal{N}$.

Appendix F Robust R-cuts

This appendix details the robust extension of the reachability cuts (R-cuts) introduced by Lysgaard [27], summarized in Section 4.3.

Let $A_U^-(l)$ be the set of arcs that can be traversed on a path from the depot to customer l without violating any time window, under the worst-case realization in U. An arc (i,j) belongs to $A_U^-(l)$ if and only if:

1. $\max\{w_0^a + S^U_{(0,i)\cup(i,j)}, w_i^a + t_{ij}\} \le w_j^b,\ t_{ij} \in U$;
2. $\max\{w_0^a + S^U_{(0,i)\cup(i,j)\cup(j,l)}, w_i^a + S^U_{(i,j)\cup(j,l)}, w_j^a + S^U_{(j,l)}\} \le w_l^b$.

These conditions mirror the deterministic construction of Lysgaard [27], with each travel time or path duration replaced by its worst-case counterpart under U (S_p^U). The set $A_U^+(l)$, used for paths from customer l back to the depot, is defined symmetrically. Given these sets, the reachability cut for a set $\mathcal{S} \subset \mathcal{N}^*$ and customer l is

$$\sum_{\substack{(i,j)\in A_U^-(l)\\ i\notin\mathcal{S},\ j\in\mathcal{S}}} x_{ij} \ge 1. \tag{49}$$

Following Lysgaard et al. [25], the separation of cut (49) for a given solution x^* is based on a max-flow (Edmonds–Karp) algorithm, applied for each node $l \in \mathcal{N}^*$ on a directed graph whose vertices are the instance's nodes, with arc capacities x^*_{ij} if $(i,j) \in A_U^-(l)$ and 0 otherwise. Computing the maximum flow between the depot 0 and node l, and subtracting it from the right-hand side of (49), gives the violation of the corresponding constraint; the node l with the greatest violation is selected, with $\mathcal{S}$ taken as the set of nodes visited in the max-flow solution. If every node has zero violation, no cut is returned. The same procedure applies to $A_U^+(l)$, computing maximum flow between customer l and the depot $n+1$.

Appendix G Results of the Branch-and-cut method.

Table 7, whose structure is similar to that of Table 3, summarizes the results of the proposed BC method for each uncertainty set and budget value. The only difference between this table and the one reporting the results of the BPC algorithm is that it includes two different columns: one for the cuts generated by CVRPSep, such as capacity and comb inequalities, and another for the robust fractional reachability cuts (R-Cuts), which are not used in the BPC algorithm.

Table 7: Computational results of the BC algorithm for the RVRPTW instances

			Solved				Cuts		
Uncertainty set	nN	Ins	nS	nS(%)	T(s)	Gap(%)	CVRPSep	Tourn	R-Cuts
Deterministic	25	56	54	96.4	240.67	0.4	36.7	0.0	0.0
	50	56	25	44.6	2063.4	5.4	112.9	0.0	0.0
	100	56	18	32.1	2502.0	12.1	545.5	0.0	0.0
Cardinality	25	504	465	92.3	348.8	0.9	32.3	38948.5	64.3
	50	504	226	44.8	2071.4	8.8	68.3	42550.5	291.4
	100	504	164	32.5	2546.3	18.4	119.5	53823.8	573.0
Knapsack	25	504	476	94.4	253.8	0.6	35.9	36096.9	4.5
	50	504	216	42.9	2111.7	6.7	109.7	54931.7	19.7
	100	504	144	28.6	2649.2	18.4	219.7	14996.6	99.8
Ellipsoidal	25	672	634	94.3	255.7	0.6	35.0	53869.6	11.7
	50	672	310	46.1	2047.5	6.9	93.1	90142.4	76.6
	100	672	209	31.1	2597.8	18.2	173.8	39543.6	259.1
Factor model	25	840	772	91.9	357.5	0.9	31.3	48202.8	66.9
	50	840	371	44.2	2085.0	9.3	73.9	54704.1	375.1
	100	840	252	30.0	2646.1	18.1	86.8	86900.0	766.5
Discrete	25	672	630	93.8	394.8	0.8	36.7	13575.0	0.2
	50	672	261	38.8	2313.6	7.9	115.9	7560.3	0.4
	100	672	147	21.9	2880.3	19.0	537.3	1659.5	0.2

Appendix H Detailed results for the robustness analysis

Tables 8 and 9 show the average results for each configuration of uncertainty set, budget, and maximum deviation over the nominal value of 10%, 25%, and 50% (*Dev*). The deterministic results are shown under column *Det*. Table 8 presents the average percentage of infeasible scenarios in the Monte Carlo simulation (Risk) for each configuration. Similarly, Table 9 presents the average price-of-robustness (PoR), which constitutes the percentage increase in costs of the robust solution over the deterministic one, for each combination of budget and deviation level. All values are presented as percentages.

Table 8: Average probability of constraint violation (Risk) of the solutions obtained for each uncertainty set considering different deviation levels in the RVRPTW instances.

	Risk (%)									
	Det	U_Ψ					U_D			
Dev	-	0	0.25	0.5	0.75	1	10	20	30	40
10%	29.91	18.73	5.60	2.95	0.24	0.0	19.05	6.13	6.58	3.57
25%	45.01	29.49	17.58	3.79	0.89	0.0	21.01	9.65	7.83	6.78
50%	54.88	40.40	22.37	7.42	1.49	0.0	22.85	13.90	10.78	8.89
	U_Γ			U_Δ			U_Ω			
Dev	1	5	10	20	40	60	0.25	0.5	0.75	1
10%	19.46	0.005	0.00	0.00	0.00	0.00	25.14	23.37	14.59	10.98
25%	22.74	0.006	0.00	0.02	0.00	0.00	30.43	25.31	22.00	12.73
50%	20.55	0.091	0.03	4.46	0.00	0.00	32.78	28.64	20.36	13.38

Appendix I Numerical information for the deterministic solution of instance R112 with 25 customers.

Table 10 reports the detailed deterministic solution for instance R112 with 25 customers. For each route, it provides the sequence of traversed arcs, the nominal travel times (t_{ij}), the maximum travel time deviations ($\hat{t}_{ij}$), and the customer time windows ($[w_j^a, w_j^b]$) at each customer.

Table 9: Average price-of-robustness (PoR) of the solutions obtained for each uncertainty set considering different deviation levels in the RVRPTW instances.

	PoR (%)									
	Det	U_Ψ					U_D			
Dev	-	0	0.25	0.5	0.75	1	10	20	30	40
10%	0.0	5.76	6.36	6.60	6.65	7.06	3.52	4.00	4.62	4.55
25%	0.0	6.76	7.91	9.07	9.23	9.63	4.27	4.33	5.04	5.61
50%	0.0	9.12	11.56	12.89	13.85	16.18	5.27	5.42	6.18	7.63
	U_Γ			U_Δ			U_Ω			
Dev	1	5	10	20	40	60	0.25	0.5	0.75	1
10%	4.69	5.35	5.44	6.14	6.15	5.97	4.53	4.71	4.95	4.98
25%	5.24	7.70	8.01	8.37	8.17	8.94	4.69	5.24	5.83	5.76
50%	8.64	13.58	14.09	10.69	14.95	15.12	6.28	6.56	7.32	9.68

Table 10: Deterministic solution for the instance R112 with 25 customers

Route 1: (Blue)	Arc	(0, 18)	(18, 8)	(8, 7)	(7, 19)	(19, 11)	(11, 10)	(10, 0)	-	-
	t_{ij}	15.8	10.4	12.2	11.1	7	11.1	25.4	-	-
	$\hat{t}_{ij}$	7.9	5.2	6.1	5.5	3.5	5.5	12.7	-	-
	$[w_j^a, w_j^b]$	[47, 136]	[50, 149]	[36, 135]	[32, 146]	[33, 152]	[85, 172]	[0, 230]	-	-
Route 2: (Green)	Arc	(0, 12)	(12, 3)	(3, 9)	(9, 20)	(20, 1)	(1, 0)	-	-	-
	t_{ij}	15	11.1	15	11.1	16.4	15.2	-	-	-
	$\hat{t}_{ij}$	7.5	5.5	7.5	5.5	8.2	7.6	-	-	-
	$[w_j^a, w_j^b]$	[15, 133]	[76, 165]	[47, 156]	[79, 182]	[73, 204]	[0, 230]	-	-	-
Route 3: (Orange)	Arc	(0, 21)	(21, 22)	(22, 23)	(23, 4)	(4, 25)	(25, 24)	(24, 0)	-	-
	t_{ij}	18	10	11.1	15	10	15	30	-	-
	$\hat{t}_{ij}$	9	5	5.5	7.5	5	7.5	15	-	-
	$[w_j^a, w_j^b]$	[18, 136]	[50, 165]	[36, 156]	[73, 182]	[56, 204]	[58, 230]	[0, 230]	-	-
Route 4: (Red)	Arc	(0, 2)	(2, 15)	(15, 14)	(14, 16)	(16, 17)	(17, 5)	(5, 6)	(6, 13)	(13, 0)
	t_{ij}	18	13	15.8	11.1	11.1	10	10	7	11.1
	$\hat{t}_{ij}$	9	6.5	7.9	5.5	5.5	5	5	3.5	5.5
	$[w_j^a, w_j^b]$	[18, 147]	[30, 152]	[32, 187]	[29, 139]	[60, 189]	[20, 167]	[49, 158]	[79, 208]	[0, 230]