Submitted to *INFORMS Journal on Computing*
manuscript JOC-2021-07-OA-180.R2

Robust Team Orienteering Problem with Decreasing Profits

Qinxiao Yu
Economics and Management College, Civil Aviation University of China, Tianjin 300300, China
qxyu@cauc.edu.cn

Chun Cheng*
Institute of Supply Chain Analytics, Dongbei University of Finance and Economics, Dalian 116025, China
*Corresponding author. chun.cheng@polymtl.ca

Ning Zhu
Institute of Systems Engineering, College of Management and Economics, Tianjin University, Tianjin 300072, China
zhuning@tju.edu.cn

This paper studies a robust variant of the team orienteering problem with decreasing profits (TOP-DP), where a fleet of vehicles are dispatched to serve customers with decreasing profits in a limited time horizon. The service times at customers are assumed to be uncertain, which are characterized by a budgeted uncertainty set. Our goal is to determine the set of customers to be served and the routes for the vehicles such that the collected profit is maximized; meanwhile, all the planned routes remain feasible for any realization of service times within the uncertainty set. We propose a two-index robust formulation for the problem, which is defined using constraints based on dynamic programming recursive equations and can be directly solved by a general-purpose optimization solver. We also present a route-based formulation for the problem, which is solved by a tailored branch-and-price (B&P) algorithm. To tackle large-size instances efficiently, we further implement a tabu search (TS) algorithm. Numerical tests show that our B&P algorithm can solve most instances with 100 customers to optimality within 30 minutes and that the TS algorithm can find high-quality solutions within a few seconds. Moreover, we find that in most cases, the robust solutions can significantly reduce the probability of deadline violations in simulation tests with only a slight compromise of profit, compared to the solutions generated by the deterministic model.

Key words: Team orienteering; uncertain service time; robust optimization; branch-and-price; tabu search
History: Accepted to *INFORMS Journal on Computing* on 12 August 2022.

1. Introduction

As a class of routing problems, the orienteering problem (OP) determines a single route to maximize the overall profit collected from selected customers within a time limit, which is usually not long enough for visiting all customers. A natural extension of the OP is the team orienteering problem (TOP), which generalizes the OP to a multi-route case (Chao et al. 1996). Namely, the TOP determines routes for multiple vehicles, each with one route, to

maximize the collected profit. As a variant of the TOP, the team orienteering problem with decreasing profits (TOP-DP) further assumes that the profit associated with each customer is decreasing with time instead of being fixed as in the TOP. Thus, any feasible route in the TOP-DP must respect the deadline constraints at customers, i.e., all selected customers must be served before their profits vanish. The TOP-DP has many practical applications, e.g., the maintenance scheduling problem, where the loss of rewards resulting from the unavailability of equipment increases with time (Tang et al. 2007). Another example is the search and rescue (SAR) operations in the relief effort, where the survival rate of trapped people declines rapidly over time (Ekici and Retharekar 2013, Afsar and Labadie 2013, Dewilde et al. 2013, Yu et al. 2022).

In the deterministic TOP-DP, parameters like travel times, service times, and customer profits are assumed to be perfectly known at the time of making decisions. However, the assumption of deterministic data is often difficult to be justified due to the fact that uncertainties commonly exist in real-world applications, like uncertain travel times resulting from traffic congestion, uncertain waiting times caused by queuing, and uncertain service times due to limited information of a service object. Indeed, in most cases, only rough information of parameters is available at the planning phase, and the precise values of parameters are gradually revealed during the operational process. Thus, it is important to proactively take uncertainties into account to guarantee the feasibility of routing decisions during the operational stage. As the profits at customers are time-sensitive in the TOP-DP, perturbations in time-related factors like the service times may cause decisions generated by the deterministic model to be infeasible. Specifically, suppose the actual service times at some selected customers are longer than the estimated values. In that case, the service at other customers might be postponed, and the route duration might be prolonged, leading to the violations of deadlines at customers and the depot. Thus, when making customer selection and vehicle routing decisions for the TOP-DP, we should pay attention to the randomness of time-related parameters. In particular, in this study, we focus on uncertain service times, which heavily depend on customers' conditions. Different conditions (e.g., the extent of equipment damage in the maintenance scheduling problem and the extent of a victim's injury in disaster scenarios) lead to different service times; thus, we cannot set a deterministic value for the service time at a customer. Moreover, although the TOP-DP

with uncertain service times commonly exists in real-world applications, it has not been addressed in the literature.

To tackle optimization problems under uncertainty, we can utilize the stochastic programming (SP) method or the robust optimization (RO) method. Under the SP framework, random variables are assumed to follow a known probability distribution, and the objective is to minimize (maximize) the expected cost (profit). This modeling paradigm is widely used to address the (T)OP under uncertainty (Tang and Miller-Hooks 2005, İlhan et al. 2008, Campbell et al. 2011, Evers et al. 2014c, Verbeeck et al. 2016, Angelelli et al. 2017, Varakantham et al. 2018, Song et al. 2020). However, the true probability distribution is rarely known in practice and is often estimated from historical samples. Solutions generated by stochastic models may lead to disappointing results when the true distribution deviates from the estimated distribution (Smith and Winkler 2006), which typically happens when the sample size is small. As an alternative tool to address uncertainties, RO does not require the knowledge of probability distribution; instead, it assumes that uncertain parameters lie in an uncertainty set with some structures, e.g., ellipsoid or polyhedron. Then optimization is performed concerning the worst-case scenario within the uncertainty set. In practical applications, it is often difficult to identify a probability distribution for the uncertain service times due to limited data or information. For example, in the context of disaster relief operations, a decision-maker has to estimate the service time required at each site for the SAR operations before planning routes. For this purpose, the decision-maker should have information about the locations of victims, their extent of injury, the structure of damaged buildings, and the capabilities of rescue teams. Whereas these pieces of information are often not perfectly known when making decisions, the decision-maker can only make a rough estimation (INSARAG 2020). Thus, we adopt an RO method for our problem, where only partial information is required. Moreover, the robust objective function, i.e., the worst-case profit, can well represent decision makers' risk preferences for some applications of the TOP-DP.

In this paper, we define a robust team orienteering problem with decreasing profits (RTOP-DP), where a budget uncertainty set characterizes the uncertain service times at customers. The RTOP-DP aims to maximize the collected profit by scheduling one route for each vehicle and also guarantee the feasibility of planned routes for any uncertainty realization within the uncertainty set. When using the RO paradigm to model the TOP-DP with

uncertain service times, the worst-case arrival time must be explicitly expressed for each visited node, to check the deadline constraints and calculate the collected profits. These arrival time-related expressions will significantly increase the number of constraints in the robust counterpart model if the classical dualization scheme is applied, making it more complex than the TOP-DP model that is already NP-hard. To explicitly incorporate uncertainties into the mathematical model whereas not significantly increase the computational complexity, we propose a tractable model for the RTOP-DP, which employs constraints based on dynamic programming recursive equations introduced by Munari et al. (2019).

1.1. Our Contributions

Our work makes the following contributions to the literature:

- We formally introduce the RTOP-DP, a new variant of the TOP-DP, by adopting an RO approach to incorporate uncertain service times.
- We first present a two-index vehicle flow model for the RTOP-DP by using constraints based on dynamic programming recursive equations. We then construct a route-based formulation for the problem and develop a branch-and-price (B&P) algorithm to solve it. To efficiently tackle the pricing problem in the B&P framework, a tailored labeling algorithm is proposed to solve the robust shortest path problem with resource constraints. We emphasize that our work is the first to solve the robust variants of the OP with a tailored exact approach. We further develop a tabu search (TS) heuristic to solve large-size instances efficiently.
- We conduct extensive numerical tests to evaluate the algorithms and the robust model. Results show that the B&P algorithm can solve most instances with 100 customers to optimality within 30 minutes and that the TS algorithm can find high-quality solutions within a few seconds. Results also show that in most cases, the robust solutions can significantly reduce the probability of deadline violations in simulation tests with only a slight compromise of profit, compared to the solutions produced by the deterministic model.

The remainder of this paper is organized as follows. Section 2 reviews related studies. The RTOP-DP is formally described in Section 3, where a robust model is constructed. Sections 4 and 5 introduce the B&P algorithm and the TS heuristic, respectively. Section 6 discusses the computational results. Finally, we conclude this paper in Section 7.

2. Literature Review

Regarding the (T)OP with time-decreasing profits, as all related studies solve deterministic problems, we extend our literature review to the OP with uncertainty. A summary of papers is provided in Table 1.

In the OP, the uncertain parameters can be mainly classified into two categories: uncertain profit and uncertain time-related factors (i.e., travel time, service time, and waiting time). İlhan et al. (2008) is the first to study an OP with uncertainty, where profits are assumed to follow normal distributions. Their objective is to maximize the probability of collecting a targeted profit level within a given time interval. The authors propose an exact branch-and-cut (B&C) method for solving small-size instances and a genetic algorithm for solving large-size instances. Angelelli et al. (2017) define a probabilistic OP, where the availabilities of customers are random and are modeled as Bernoulli variables. A recourse action is allowed to skip absent customers when their availabilities are revealed. The authors propose a two-stage SP model, which is solved by a B&C method. Heuristic strategies are further developed to reduce the search space of the B&C, leading to a matheuristic method. Similarly, Song et al. (2020) also assume that the service requests of customers are stochastic in the context of TOP. They first develop a two-stage SP model for the problem and then introduce a customer assignment policy where a set of scenarios are sampled. Finally, they develop a B&P algorithm to solve the deterministic TOP in each scenario. From these three papers, we find that the uncertain profit only affects the optimal value of a model, and does not affect the feasibility of its solution, because each planned route is only subject to time constraints while all time-related parameters are unchanged.

As perturbations of time-related parameters may lead to infeasible solutions, it is more challenging to address problems with uncertain times. Tang and Miller-Hooks (2005) incorporate the uncertain service times into the OP, which are assumed to have discrete probability distributions. They formulate this problem as a chance-constrained model to maximize the total profit as well as to ensure that the duration of each route meets the constraint with a predefined probability. Campbell et al. (2011) introduce the OP with stochastic travel and service times (they call it OPSTS for short) and assume that the distributions of random variables are independent and identical. The authors propose an exact approach based on dynamic programming (DP) and a variable neighborhood search

(VNS) method to solve the problem. Papapanagiotou et al. (2016) study the same problem but focus on developing a more efficient objective function evaluator via combining Monte Carlo sampling and an analytical solution to speed up the VNS. Another paper addressing the OPSTS is by Bian and Liu (2018), where the authors propose a real-time adjustment strategy called simulation-aided multiple plan approach. In particular, the Monte Carlo simulation is used to evaluate the in-time arrival probability, and a multi-plan approach consisting of a TS and a myopia prevention strategy is employed to generate and update the real-time solution. Evers et al. (2014c) develop a two-stage SP model for the OP with stochastic weights (OPSW), where the weights refer to the travel times. They apply the sample average approximation (SAA) for solving small-size instances and a heuristic algorithm for solving large-size instances. Evers et al. (2014b) use an RO approach to model the OPSW, where the weights refer to the fuel usage between targets. They solve the robust counterpart model using CPLEX and conduct experiments on the instances with 20 nodes. To the best of our knowledge, Evers et al. (2014b) are the first to use the RO technique for the OP with uncertainty. The OPSW is also studied by Dolinskaya et al. (2018), where a VNS method incorporating a DP model is used to select reward nodes and decide paths.

Jin and Thomas (2019) study a TOP with both uncertain service times and profits, where these uncertainties are governed by a queueing process. The authors propose a local search algorithm to provide a priori solution with the maximum expected profit. Balcik and Yanıkoğlu (2020) focus on a rapid needs assessment planning problem after a disaster, which can be formulated as a TOP with uncertain travel times. They present an RO model with a coaxial box uncertainty for the problem and develop a practical method for evaluating route feasibility. A TS algorithm is proposed to solve the model. Varakantham et al. (2018) and Liao and Zheng (2018) both study the time-dependent OP (TD-OP), where the uncertain time-related factors follow different types of distribution functions in different time slots. The former proposes an SAA approach and a local search method to solve the chance-constrained SP model, and the latter applies a hybrid heuristic algorithm.

Since the deadline can be viewed as a special type of time window, the TOP-DP is a special case of the TOP with time windows (TOPTW). Thus, the most related problem to the RTOP-DP is the uncertain variant of the TOPTW. However, the TOPTW with uncertainty has not been addressed in the literature, and only the OPTW (i.e., the single-vehicle version) with uncertainty has been studied by some authors. Evers et al. (2014a)

Table 1 A Summary of Papers on Orienteering Problems Under Uncertainty

Authors	Problem	Uncertain Parameter				Modeling Framework	Solution Method	
		Travel Time	Service Time	Wait Time	Profit		Exact	Heuristic
İlhan et al. (2008)	OP				✓	SP	B&C	Genetic algorithm
Angelelli et al. (2017)	OP				✓	SP	B&C	Matheuristic algorithm
Song et al. (2020)	TOP				✓	SP	B&P	Heuristic policy
Tang and Miller-Hooks (2005)	OP		✓			SP	B&C	Construct-and-adjust
Campbell et al. (2011)	OP	✓	✓			SP	DP	Variable neighborhood search
Papapanagiotou et al. (2016)	OP	✓	✓					Variable neighborhood search
Bian and Liu (2018)	OP	✓	✓					Simulation-aided multi-plan approach
Evers et al. (2014c)	OP	✓				SP		Construct-and-improve
Evers et al. (2014b)	OP	✓				RO	Solver	
Dolinskaya et al. (2018)	OP	✓				DP		Variable neighborhood search
Jin and Thomas (2019)	TOP		✓		✓	Queuing theory		Local search
Balcik and Yanıkoğlu (2020)	TOP	✓				RO		Tabu search
Varakantham et al. (2018)	TD-OP	✓				SP		Local search
Liao and Zheng (2018)	TD-OP	✓		✓				Hybrid heuristic
Evers et al. (2014a)	OPTW	✓	✓			Re-planning		Local search
Zhang et al. (2014)	OPTW			✓		Queueing theory		Variable neighborhood search
Zhang et al. (2018)	OPTW			✓		MDP		Approximate dynamic programming
Zhang et al. (2020)	MP-OPTW			✓		MDP		Assign-and-routing heuristic
Verbeeck et al. (2016)	TD-OPTW	✓				SP		Ant colony optimization
This work	TOP-DP		✓			RO	B&P	Tabu search

are the first to consider the OPTW with uncertain travel and service times. They propose an online approach where a re-planning model maximizing the expected profit is called each time before leaving a customer. A fast heuristic is developed to solve the re-planning model. Zhang et al. (2014) suggest that queueing at customers leads to stochastic waiting times and address the OPTW with uncertain waiting times. They derive an analytical formula to compute the expected profit from an a priori tour and employ a VNS to solve the problem. For the same problem, Zhang et al. (2018) model it as a Markov decision process (MDP) to maximize the expected profit. They propose an approximate DP approach for the problem. Subsequently, Zhang et al. (2020) study a multi-period OPTW (MP-OPTW) and also use the MDP to model the problem. Verbeeck et al. (2016) consider the time-dependent OPTW (TD-OPTW) and assume that the travel time is a stochastic function depending on the departure time at a predecessor customer. A stochastic ant colony system is used to solve the problem.

From the reviewed papers and Table 1, we can see that growing attention has been drawn to the OP with uncertainty, where different types of uncertain factors are considered and a variety of modeling and solution methods are developed. Whereas almost all the studies assume that the probability distribution of random variables is perfectly known, and only Evers et al. (2014b) and Balcik and Yanıkoğlu (2020) are the exceptions where exact probability information is not required, which is typically the case in practice. Moreover, all existing studies on the OPTW under uncertainty propose heuristic methods for solving

their models. The reason might be that it is easy to check the feasibility of time windows for a given route by using sampling or simulation in heuristic algorithms. However, in an exact method framework, to check the time window constraints, we must track the arrival times at all nodes, which requires a large number of constraints in a compact formulation. When uncertainties are further incorporated, the number of arrival time-related constraints increases significantly if the classical dualization scheme is applied to reformulate the robust model as in the literature, making the problem even harder to solve by exact algorithms. To overcome these challenges, we use constraints based on dynamic programming recursive equations, which help us to construct a compact RTOP-DP model and develop the B&P algorithm.

3. Mathematical Model

This section describes the problem and presents the deterministic and the robust models.

3.1. Problem Description

The TOP-DP is defined on a complete directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = \mathcal{N}_c \cup \{0\}$ is the set of nodes, including the subset of customers $\mathcal{N}_c = \{1, \ldots, n\}$ and the depot 0. $\mathcal{A} = \{(i,j) : i,j \in \mathcal{N}, i \neq j\}$ is the set of arcs. The travel time on arc $(i,j) \in \mathcal{A}$ is denoted by t_{ij}. As in most studies on the TOP-DP (Ekici and Retharekar 2013, Afsar and Labadie 2013, Dewilde et al. 2013, Yu et al. 2022), we assume the profit p_i of customer i decreases linearly with time and the decay ratio is d_i. Thus, the service at customer i can only be conducted during the time interval where the customer is still profitable, i.e., the deadline of serving customer i is $D_i = p_i/d_i$. We assume that the service time at customer i is s_i. There is no profit associated with the depot. We consider a fleet of homogeneous vehicles $\mathcal{K} = \{1, \ldots, K\}$, which start from the depot to serve assigned customers and finally return to the depot within a given time interval $[0, T_{\max}]$. Then the deadline at the depot or the time horizon of the considered problem is $T_{\max}$. The goal of the TOP-DP is to determine K routes serving a subset of customers such that the collected profit is maximized. To this end, we make two types of decisions: the assignment of customers to vehicles and the service sequence of customers along each route. To model the TOP-DP, we adopt a two-index vehicle flow formulation, as Munari et al. (2019) indicate that the robust counterpart of a two-index formulation for the vehicle routing problem (VRP) with time windows can be optimally solved by a general-purpose optimization solver like CPLEX and Gurobi

in reasonable computing times for instances with 25 or more customers. We define the following decision variables:

- x_{ij}: a binary variable equalling to 1 if node $j \in \mathcal{N}$ is visited immediately after node $i \in \mathcal{N}$, 0 otherwise;
- y_i: a binary variable equalling to 1 if customer $i \in \mathcal{N}_c$ is served, 0 otherwise;
- a_i: a continuous variable indicating the arrival time at node $i \in \mathcal{N}$.

3.2. Deterministic Model

The deterministic TOP-DP is formulated as follows:

$$\max \sum_{i \in \mathcal{N}_c} p_i y_i - d_i a_i \tag{1}$$

$$\text{s.t.} \sum_{j \in \mathcal{N}_c} x_{0j} = K, \tag{2}$$

$$\sum_{i \in \mathcal{N}_c} x_{i0} = K, \tag{3}$$

$$\sum_{j \in \mathcal{N}, j \neq i} x_{ij} = \sum_{j \in \mathcal{N}, j \neq i} x_{ji} \qquad \forall i \in \mathcal{N}_c, \tag{4}$$

$$\sum_{j \in \mathcal{N}, j \neq i} x_{ji} = y_i \qquad \forall i \in \mathcal{N}_c, \tag{5}$$

$$a_i + s_i + t_{ij} \leq a_j + M_{ij}(1 - x_{ij}) \qquad \forall i \in \mathcal{N}_c, j \in \mathcal{N}, \tag{6}$$

$$a_i \geq t_{0i} y_i \qquad \forall i \in \mathcal{N}_c, \tag{7}$$

$$a_i \leq D_i y_i \qquad \forall i \in \mathcal{N}_c, \tag{8}$$

$$a_0 \leq T_{\max}, \tag{9}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in \mathcal{N}, \tag{10}$$

$$y_i \in \{0, 1\} \qquad \forall i \in \mathcal{N}_c, \tag{11}$$

$$a_i \geq 0 \qquad \forall i \in \mathcal{N}. \tag{12}$$

The objective function (1) maximizes the overall profit collected by all the vehicles. Constraints (2)–(3) ensure that exactly K vehicles depart from and return to the depot. Constraints (4) guarantee the flow conservation at each customer. Constraints (5) ensure that variable y_i is uniquely determined by variables x_{ij} and that each customer is served at most once. Constraints (6) are the Miller-Tucker-Zemlin (MTZ) constraints, which describe the relationship of arrival times between two adjacent nodes (Miller et al. 1960). We set the large constants $M_{ij} = D_i + s_i + t_{ij}$. Note that constraints (6) are not defined for arcs

$(0,i)$, $i \in \mathcal{N}_c$. Constraints (7) are consequently used to define the earliest arrival time at a served customer, which is larger than or equal to the travel time from the depot to the customer. Constraints (8) indicate that the latest arrival time at a served customer must not be later than its deadline. If a customer is not served, constraints (7) and (8) enforce its arrival time to be 0. Constraints (9) ensure that all the vehicles return to the depot no later than $T_{\max}$. Constraints (10)–(12) define the domains of decision variables.

3.3. Robust Model

In this section, we first introduce the uncertainty set and then construct the robust model.

3.3.1. Uncertainty Set. We assume that the uncertain service times are independent random variables falling within a range of estimated values denoted as

$$\tilde{s}_i = \bar{s}_i + \xi_i \hat{s}_i, \xi_i \in [-1,1], \forall i \in \mathcal{N}_c, \tag{13}$$

where $\bar{s}_i$ is the nominal service time at customer i; $\hat{s}_i$ is the largest deviation of service time from its nominal value; and ξ_i is the uncertain parameter belonging to $[-1,1]$. We employ the budgeted uncertainty set proposed by Bertsimas and Sim (2004) to represent randomness and control the conservatism of robust solutions, which is written as

$$\mathcal{U} = \left\{ \tilde{\boldsymbol{s}} \in \mathbb{R}^{|\mathcal{N}_c|} : \tilde{s}_i = \bar{s}_i + \xi_i \hat{s}_i,\ \sum_{i \in \mathcal{N}_c} \xi_i \le \Gamma,\ 0 \le \xi_i \le 1, \forall i \in \mathcal{N}_c \right\}, \tag{14}$$

where parameter Γ is called the *uncertainty budget*, which takes an integer value in the interval $[0, |\mathcal{N}_c|]$. Γ controls the level of conservatism by limiting the number of customers whose service times are allowed to deviate from the nominal values. Although the uncertain parameter ξ_i is defined to be in the interval $[-1,1]$ in equation (13), considering that the worst-case scenario is realized only when service times have positive deviations, it is reasonable to assume that ξ_i belongs to $[0,1]$ in the uncertainty set $\mathcal{U}$.

We are aware that other types of uncertainty sets can be utilized to characterize the uncertain service times, like the box and the ellipsoidal uncertainty sets. However, the box uncertainty set is very conservative because it allows all the uncertain parameters $\xi_i, i \in \mathcal{N}_c$ to take on their worst-case values simultaneously. Although the ellipsoidal uncertainty set is less conservative than the box uncertainty set, it would lead to a nonlinear model due to the nonlinear constraint for defining the ellipsoid in the uncertainty set. Thus, we use the budgeted uncertainty set for our problem, which can simultaneously control the conservatism of robust solutions (via adjusting the value of Γ) and generate a tractable robust

model. In practical applications, the uncertainty budget Γ can be calibrated according to decision-makers' risk preferences or prior experiences. It can also be justified by sensitivity analysis and simulation experiments (see Section 6.5), i.e., we present decision-makers with a set of solutions by varying the value of Γ, each achieving different levels of compromises between the profit and the robustness. Then they can decide the value of Γ based on the profit-robustness trade-off.

3.3.2. Robust Formulation. To construct the robust model, we further introduce another continuous decision variable $a_{i\gamma}$, which indicates the arrival time at node $i \in \mathcal{N}$ when $\gamma \in \{0, 1, \ldots, \Gamma\}$ predecessors' service times have reached their worst-case values. Before formally presenting the RO model, we first introduce the approach for calculating the worst-case arrival time at each customer.

Let $r = (v_0, v_1, v_2, \ldots, v_m, v_{m+1})$ be a route that starts from the depot $v_0 = 0$, sequentially serves m customers, and finally returns to the depot $v_{m+1} = 0$. In the context of RO, route r is feasible only if the following conditions are satisfied for all the realizations of uncertain service times in $\mathcal{U}$: (i) any customer along this route is served at most once before its deadline, and (ii) the vehicle returns to the depot v_{m+1} no later than the deadline $T_{\max}$. To check the robust feasibility of route r, the arrival time at each node must be explicitly expressed in the worst-case realization of the uncertainty set. Based on the structure of the budgeted uncertainty set $\mathcal{U}$, the arrival time can be efficiently computed using dynamic programming recursive equations (Agra et al. 2013, Munari et al. 2019). Specifically, let $a_{v_j\gamma}$ be the earliest arrival time at node v_j of a given route when at most $\gamma \le \Gamma$ customers' service times have reached their worst-case values. Note that we call γ as *uncertainty tolerance* in the rest of this paper for easy of description. Then $a_{v_j\gamma}$ can be computed by the following recursive equation:

$$a_{v_j\gamma} = \begin{cases} 0, & j = 0,\ 0 \le \gamma \le \Gamma, \quad (15a) \\ t_{v_0 v_j}, & j = 1,\ 0 \le \gamma \le \Gamma, \quad (15b) \\ a_{v_{j-1}\gamma} + \bar{s}_{v_{j-1}} + t_{v_{j-1}v_j}, & j > 1,\ \gamma = 0, \quad (15c) \\ \max\Big\{ a_{v_{j-1}\gamma} + \bar{s}_{v_{j-1}} + t_{v_{j-1}v_j}, & \\ \quad a_{v_{j-1}\gamma-1} + \bar{s}_{v_{j-1}} + \hat{s}_{v_{j-1}} + t_{v_{j-1}v_j} \Big\}, & j > 1,\ 0 < \gamma \le \Gamma. \quad (15d) \end{cases}$$

Case (15a) indicates that each vehicle starts its trip form the depot v_0 at time 0. As no service is required at the depot, each vehicle arrives at its first customer v_1 at time $t_{v_0 v_1}$

regardless of the value of γ, leading to case (15b). For the next few nodes $v_j, j > 1$ along the route, there exist two cases for calculating the arrival time. When $\gamma = 0$, i.e., the service times at all customers before v_j take their nominal values and none of them reach their worst-case values, the arrival time at v_j is equal to the arrival time at its predecessor (i.e., $a_{v_{j-1}}$) plus the nominal service time there (i.e., $\bar{s}_{v_{j-1}}$) and the travel time $t_{v_{j-1}v_j}$ between them, leading to case (15c). When $0 < \gamma \leq \Gamma$, i.e., there are γ served customers (before v_j) that have attained their worst-case service times, we need to consider two situations with respect to the composition of the γ customers for calculating the value of $a_{v_j\gamma}$. The first situation is that there are γ customers attaining their worst-case service times before v_{j-1}. And the second one is that there are $\gamma - 1$ customers reaching their worst-case service times before v_{j-1} and the next customer with the worst-case service time would be v_j. Accordingly, $a_{v_j\gamma}$ is computed in the same way as in case (15c) for the former situation, and $a_{v_j\gamma} = a_{v_{j-1}\gamma-1} + \bar{s}_{v_{j-1}} + \hat{s}_{v_{j-1}} + t_{v_{j-1}v_j}$ for the latter situation. Finally, $a_{v_j\gamma}$ takes the maximum value between the results generated by the two situations as indicated in equation (15d). Since $a_{v_0\gamma}$ and $a_{v_1\gamma}$ are known for all $\gamma \leq \Gamma$, we can use the recursive equation (15) to find $a_{v_j\gamma}$ for all $j > 1$ and $0 \leq \gamma \leq \Gamma$. To be robust feasible, each route must satisfy $a_{v_j\Gamma} \leq D_{v_j}$ for all $j = 1, \ldots, m+1$. Note that $D_{v_{m+1}} = T_{\max}$ for the depot $v_{m+1} = 0$.

Now we can formulate the RTOP-DP as

$$\max \sum_{i \in \mathcal{N}_c} p_i y_i - d_i a_{i\Gamma} \tag{16}$$

s.t. Constraints (2)–(5) and (10)–(11),

$$a_{i\gamma} + \bar{s}_i + t_{ij} \leq a_{j\gamma} + M'_{ij}(1 - x_{ij}) \qquad \forall i \in \mathcal{N}_c, j \in \mathcal{N}, \gamma \in \{0, \ldots, \Gamma\}, \tag{17}$$

$$a_{i\gamma-1} + \bar{s}_i + \hat{s}_i + t_{ij} \leq a_{j\gamma} + M''_{ij}(1 - x_{ij}) \qquad \forall i \in \mathcal{N}_c, j \in \mathcal{N}, \gamma \in \{1, \ldots, \Gamma\}, \tag{18}$$

$$a_{i\gamma} \geq t_{0i} y_i \qquad \forall i \in \mathcal{N}_c, \gamma \in \{0, \ldots, \Gamma\}, \tag{19}$$

$$a_{i\gamma} \leq D_i y_i \qquad \forall i \in \mathcal{N}_c, \gamma \in \{0, \ldots, \Gamma\}, \tag{20}$$

$$a_{0\gamma} \leq T_{\max} \qquad \forall \gamma \in \{0, \ldots, \Gamma\}, \tag{21}$$

$$a_{i\gamma} \geq 0 \qquad \forall i \in \mathcal{N}, \gamma \in \{0, \ldots, \Gamma\}. \tag{22}$$

The objective function (16) maximizes the overall worst-case profit collected by all the vehicles. Based on equation (15), constraints (17) and (18) describe the relationship of arrival times between two adjacent nodes without and with one unit increment of uncertainty tolerance, respectively. Specifically, constraints (17) represent the case where the

uncertainty tolerance keeps unchanged from nodes i to j, corresponding to case (15c) and the first situation of case (15d). We set the large constants $M'_{ij} = D_i + \bar{s}_i + t_{ij}$. Constraints (18) denote the case where the uncertainty tolerance increases by one unit from nodes i to j, corresponding to the second situation of case (15d), then an additional term $\hat{s}_i$ must be added to the left-hand side of the constraints to indicate that the worst-case service time is realized at node i. We set the large constants $M''_{ij} = D_i + \bar{s}_i + \hat{s}_i + t_{ij}$. These constraints are similar to the MTZ constraints (6) in the deterministic model but defined here for all the uncertainty tolerance $\gamma \in \{0, 1, \ldots, \Gamma\}$. Munari et al. (2019) first introduce these constraints for a robust vehicle routing problem with time windows, which can model robustness directly without using the duality technique that is commonly applied in the RO literature to reformulate the robust counterpart to a tractable mathematical model. Constraints (19) and (20) indicate the lower and upper bounds of the arrival time at customer i. Constraints (21) ensure that all the vehicles return to the depot no later than $T_{\max}$ under any uncertainty tolerance. Constraints (22) define the domains of variables $a_{i\gamma}$.

As an intuitive extension of the deterministic model, the robust model is compact and can be directly solved by an optimization solver. Our preliminary experiments show that CPLEX can solve 25-customer instances with a short time horizon to optimality within one hour. However, the computing time increases significantly with the instance size and the time horizon. To solve the problem efficiently and exactly, we introduce a route-based formulation and a B&P algorithm in the next section. In addition, we note that our modeling framework introduced in this section can be directly extended to formulate the RTOP-DP with both uncertain service times and uncertain travel times. We provide the corresponding modifications to the uncertainty set, the recursive equations, and the robust model for this extension in Appendix A in the online supplement.

4. Branch-and-Price Method

This section first introduces a route-based formulation for the RTOP-DP, based on which a B&P method is developed. We then describe the master problem and the pricing problem defined in the column generation (CG) framework in Sections 4.2 and 4.3, respectively. A heuristic method for identifying integer solutions of the RTOP-DP during the B&P process is provided in Section 4.4. Finally, Section 4.5 presents the branching scheme.

4.1. A Route-based Formulation

The goal of the RTOP-DP is to find K robust feasible routes of a maximum profit such that each customer is served at most once. We define a robust feasible route as $r = (0, v_1, v_2, \ldots, v_{\mu_r}, 0)$, where μ_r is the number of served customers. Let $\mathcal{R}$ be the set of all robust feasible routes. Parameter p_r denotes the profit of route $r \in \mathcal{R}$, which is the sum of the actual profits collected from served customers. Parameter π_{ir} indicates the number of times route r serves customer i. We are interested in finding a robust feasible solution $sol = \{r_1, r_2, \ldots, r_K\}$, where the customers served by each route $r_k \in \mathcal{R}$ are exclusive and the total profit of these routes is the maximum.

Let $\theta_r \in \{0, 1\}$ be a binary variable equalling to 1 if route $r \in \mathcal{R}$ is selected, 0 otherwise. The route-based formulation [P] is constructed as follows:

$$[\text{P}]: \max \sum_{r \in \mathcal{R}} p_r \theta_r \tag{23}$$

$$\text{s.t.} \sum_{r \in \mathcal{R}} \theta_r \leq K, \tag{24}$$

$$\sum_{r \in \mathcal{R}} \pi_{ir} \theta_r \leq 1 \qquad \forall i \in \mathcal{N}_c, \tag{25}$$

$$\theta_r \in \{0, 1\} \qquad \forall r \in \mathcal{R}. \tag{26}$$

The objective function (23) maximizes the profit by selecting routes. Constraint (24) guarantees that at most K routes are selected in the final solution. We define the number of selected routes to be smaller than or equal to K, instead of being exactly K, because all the vehicles would be used in the optimal solution even with an inequality constraint. Keeping equality in constraint (24), however, would lead to a free dual variable, typically slowing down the convergence of the CG algorithm (Feillet 2010). Constraints (25) ensure that each customer is served at most once. Constraints (26) are integrality constraints.

Due to the large size of set $\mathcal{R}$, it is difficult or even impossible to enumerate all the robust feasible routes. Therefore, we use a CG algorithm to solve the linear relaxation of model [P], where routes are dynamically added. To obtain the optimal integer solution for model [P], we incorporate the CG into a branch-and-bound (B&B) framework, leading to a B&P method.

4.2. Restricted Master Problem

At each node of the B&B tree, a linear relaxation problem [LP] of model [P] is solved to optimality by the CG, which comprises a restricted master problem [RMP] and a pricing

subproblem. The [RMP] is obtained from the [LP] with a reduced subset $\hat{\mathcal{R}} \subseteq \mathcal{R}$. At the root node of the B&B tree, we initialize subset $\hat{\mathcal{R}}$ with one dummy route that visits each customer once and has a negative profit $\mathcal{P}$ (we set $\mathcal{P} = -\sum_{i \in \mathcal{N}_c} p_i$). This dummy route will be kept in $\hat{\mathcal{R}}$ in the remaining B&B procedures; however, it would not appear in the final solution due to its greatly negative profit (unless the [RMP] is infeasible). Starting from solving the [RMP] with a given $\hat{\mathcal{R}}$, the primal solution $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{|\hat{\mathcal{R}}|})$ and the dual solution $\boldsymbol{\lambda} = (\lambda_0, \lambda_1, \ldots, \lambda_{|\mathcal{N}c|})$ are obtained, where λ_0 is a nonnegative dual variable for constraint (24) and λ_i is a nonnegative dual variable associated with constraints (25) indexed by customer $i \in \mathcal{N}_c$.

If the optimal solution $\boldsymbol{\theta}$ of the [RMP] is also optimal for the [LP], then the corresponding linear relaxation problem at this node is optimally solved. Checking the optimality of the current primal solution $\boldsymbol{\theta}$ to the [LP] is equivalent to checking if the dual solution $\boldsymbol{\lambda}$ is optimal to the dual problem of the [LP]. We write the dual of the [LP] as

$$\text{[DLP]: } \min K\lambda_0 + \sum_{i \in \mathcal{N}_c} \lambda_i \tag{27}$$

$$\text{s.t. } \lambda_0 + \sum_{i \in \mathcal{N}_c} \pi_{ir} \lambda_i \geq p_r \qquad \forall r \in \mathcal{R}, \tag{28}$$

$$\lambda_i \geq 0 \qquad \forall i \in \mathcal{N}. \tag{29}$$

If there exists any robust feasible route $r' \in \mathcal{R}$ violating constraints (28), the current dual solution $\boldsymbol{\lambda}$ is not optimal for the [DLP], neither is $\boldsymbol{\theta}$ optimal for the [LP]. Moreover, it implies that moving route r' to the subset $\hat{\mathcal{R}}$ will increase the objective value of the [RMP], which is closer to the optimal value of the [LP].

4.3. Pricing Problem

Given a dual solution $\boldsymbol{\lambda}$, we define the reduced cost of a robust feasible route r as

$$c_r(\boldsymbol{\lambda}) = \lambda_0 + \sum_{i \in \mathcal{N}_c} \pi_{ir} \lambda_i - p_r. \tag{30}$$

Thus, the pricing problem is to find a robust feasible route with minimum reduced cost, that is, $\min_{r \in \mathcal{R}} c_r(\boldsymbol{\lambda})$. If the reduced cost of the found route is negative, we add this route to set $\hat{\mathcal{R}}$ and solve the [RMP] with the updated subset $\hat{\mathcal{R}}$ again. Otherwise, no route can improve the total profit any more, and the CG algorithm terminates.

Recall that the actual profit of a route in our problem is calculated as $p_r = \sum_{i \in r} p_i - d_i a_{i\Gamma}$, then the reduced cost can be rewritten as

$$c_r(\boldsymbol{\lambda}) = \lambda_0 + \sum_{i \in \mathcal{N}_c} \pi_{ir}\lambda_i - \sum_{i \in \mathcal{N}_c} (p_i - d_i a_{i\Gamma})\pi_{ir} = \lambda_0 + \sum_{i \in \mathcal{N}_c} \pi_{ir}(\lambda_i - p_i + d_i a_{i\Gamma}).$$

Let β_{ijr} denote the number of times that arc (i,j) is traversed by route r. We set $p_0 = d_0 = 0$ and further rewrite the reduced cost of route r as

$$c_r(\boldsymbol{\lambda}) = \sum_{(i,j) \in A} \beta_{ijr}(\lambda_j - p_j + d_j a_{j\Gamma}). \tag{31}$$

Thus, a modified cost $\bar{c}_{ij} = \lambda_j - p_j + d_j a_{j\Gamma}$ is assigned to arc $(i,j) \in \mathcal{A}$.

The pricing problem corresponds to the well-known elementary shortest path problem with resource constraints (ESPPRC) on graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$. The ESPPRC is NP-hard, solving it requires a significant amount of computing time (Dror 1994). When implementing the B&P algorithm, we need to solve the ESPPRC for many times. Thus, it is a common practice to solve its relaxations, that is, the requirement on elementary route is totally relaxed (i.e., SPPRC) or partially relaxed (e.g., *k*-cycle-SPPRC, *partially* ESPPRC, and *ng*-Path). We refer interested readers to Costa et al. (2019) for more details about the relaxations of the ESPPRC.

As a special case of the *k*-cycle-SPPRC, the *2*-cycle-SPPRC prohibits generating cycles with a length of 2 (e.g., *i*-*j*-*i*), which provides stronger bounds than the SPPRC. Moreover, Houck et al. (1980) also show that the *2*-cycle-SPPRC does not increase the complexity of the labeling algorithm for the SPPRC, which can be solved in a pseudo-polynomial computing time (Desrochers et al. 1992). Compared to the ESPPRC, solving the *2*-cycle-SPPRC as the pricing problem may significantly reduce the computing time of the B&P algorithm. Thus, the *2*-cycle-SPPRC has been widely used in the literature. Our preliminary numerical results (provided in Appendix B in the online supplement) also show that the *2*-cycle-SPPRC works better than the *3*-cycle-SPPRC; thus, we adopt the *2*-cycle-SPPRC as the pricing problem.

In the labeling algorithm, each partial path from the depot to a given node $i \in \mathcal{N}$ is represented by a label $\mathcal{L}_i = (pre_i, c_i, a_{i0}, \ldots, a_{i\Gamma})$, where pre_i is the direct predecessor of node i; c_i is the reduced cost up to node i; and $a_{i\gamma}$ is the arrival time at node i when at most $\gamma \in \{0, 1, \ldots, \Gamma\}$ nodes that are served before node i have achieved their worst-case service times. Starting from the depot 0, a label is initially set as $\mathcal{L}_0 = (null, 0, 0, \ldots, 0)$.

Then, the algorithm extends label $\mathcal{L}_i$ to node $j \in \mathcal{N} \setminus \{i, pre_i\}$, generating a new label $\mathcal{L}_j$ according to the following relationships:

$$pre_j = i, \tag{32a}$$

$$c_j = c_i + \lambda_j - p_j + d_j a_{j\Gamma}, \tag{32b}$$

$$a_{j0} = a_{i0} + t_{ij} + \bar{s}_i, \tag{32c}$$

$$a_{j\gamma} = \max\{a_{i\gamma} + \bar{s}_i + t_{ij}, a_{i\gamma-1} + \bar{s}_i + \hat{s}_i + t_{ij}\} \quad \forall \gamma \in \{1, \ldots, \Gamma\}. \tag{32d}$$

The resulting path associated with label $\mathcal{L}_j$ is robust feasible only if $a_{j\gamma} \leq D_j, \ \forall \gamma \in \{0, 1, \ldots, \Gamma\}$.

To speed up the labeling algorithm, a dominance rule is necessary to eliminate unpromising labels. The basic dominance rule for the *2*-cycle-SPPRC only applies to labels having the same ending and predecessor nodes, which is not efficient (Irnich and Villeneuve 2006). Kohl (1995) and Jesper (1999) propose an enhanced dominance rule for the *2*-cycle-SPPRC, where they define three types of dominant labels such that labels with the same ending node but with different predecessors could also be involved in the dominance relation. We adapt this enhanced dominance rule for our robust *2*-cycle-SPPRC as our preliminary numerical results (presented in Appendix B in the online supplement) confirm that the enhanced dominance rule performs better than the basic one.

We first define the dominance relation between two labels in the context of robust SPPRC. Given two labels $\mathcal{L}_i^1$ and $\mathcal{L}_i^2$ ending at the same node i, $\mathcal{L}_i^2$ **is dominated by** $\mathcal{L}_i^1$ if the following conditions are satisfied: (i) $c_i^1 \leq c_i^2$, and (ii) $a_{i\gamma}^1 \leq a_{i\gamma}^2, \forall \gamma \in \{0, 1, \ldots \Gamma\}$. Next, we introduce the three types of dominant labels in the enhanced dominance rule for the robust *2*-cycle-SPPRC.

DEFINITION 1. A label $\mathcal{L}_i = (pre_i, c_i, a_{i0}, \ldots, a_{i\Gamma})$ is defined as **strongly dominant** if it is not dominated by any other labels, and there exists at least one $\gamma \in \{0, \ldots, \Gamma\}$ making relation $(a_{i0} + \bar{s}_i + t_{ipre_i}) > D_{pre_i}$ or $\max\{a_{i\gamma} + \bar{s}_i + t_{ipre_i}, a_{i\gamma-1} + \bar{s}_i + \hat{s}_i + t_{ipre_i}\} > D_{pre_i}$ hold.

The two relations in definition 1 mean that under an uncertainty tolerance γ, the arrival time at node pre_i is later than its deadline D_{pre_i}, leading to an infeasible path. Thus, a strongly dominant label $\mathcal{L}_i$ cannot be extended to its predecessor pre_i. For any label (say $\mathcal{L}_i^2$) dominated by $\mathcal{L}_i$, since condition $a_{i\gamma}^2 \geq a_{i\gamma}$ holds for all $\gamma \in \{0, 1, \ldots \Gamma\}$, there must exist a γ making it infeasible for label $\mathcal{L}_i^2$ to be extended to the same predecessor as $\mathcal{L}_i$. Thus, $\mathcal{L}_i^2$ can be discarded no matter whether its predecessor is the same as or different from that of $\mathcal{L}_i$.

DEFINITION 2. A label $\mathcal{L}_i = (pre_i, c_i, a_{i0}, \ldots, a_{i\Gamma})$ is defined as **semistrongly dominant** if it is not dominated by any other labels and it is not strongly dominant.

DEFINITION 3. A label $\mathcal{L}_i = (pre_i, c_i, a_{i0}, \ldots, a_{i\Gamma})$ is defined as **weakly dominant** if it is only dominated by semistrongly dominant labels that share the identical predecessor node but not pre_i.

If a label is semistrongly or weakly dominant, it will be kept in the labeling procedure. We keep a semistrongly dominant label because it is not dominated by any other label. The reason for keeping a weakly dominant label $\mathcal{L}_i$ is explained as follows. Let pre_i^1 denote the identical predecessor shared by several semistrongly dominant labels that dominate $\mathcal{L}_i$ and $pre_i^1 \neq pre_i$. Since these semistrongly dominant labels cannot be extended to pre_i^1 due to the 2-cycle limit, their possible extension nodes belong to set $\bar{\mathcal{N}}^1 = \mathcal{N} \setminus \{i, pre_i^1\}$. For the weakly dominant label $\mathcal{L}_i$, it can only be extended to a node in set $\bar{\mathcal{N}} = \mathcal{N} \setminus \{i, pre_i\}$. Since $pre_i^1 \neq pre_i$, we have $\bar{\mathcal{N}} \not\subset \bar{\mathcal{N}}^1$. Then $\mathcal{L}_i$ cannot be discarded even if it is dominated by the aforementioned semistrongly dominant labels. Now we conclude that any label that does not belong to the aforementioned three types can be discarded in the labeling procedure.

4.4. A Heuristic for Integer Solution

After solving a node in the B&B tree by the CG algorithm, a set of routes (most include cycles) are generated by the pricing problem. If the optimal solution of a node is fractional, before branching, we try to find a feasible integer solution to the RTOP-DP from the current set of routes by imposing integrality to all variables in the [RMP]. This operation results in an integer programming model, which is always feasible and solved quickly by an optimization solver. In this way, feasible solutions sometimes can be obtained at an earlier stage of the B&P process, which helps accelerate the algorithm, because tighter bounds are found earlier (Munari et al. 2019, Santini et al. 2018, Alvarez and Munari 2017).

4.5. Branching Scheme

The branching scheme for the TOP is different from that for the VRP because it is optional to visit a customer in the TOP whereas compulsory in the VRP. Boussier et al. (2007) suggest that branching on a fractional arc is affected by the constraints of customers associated with this arc, and thus they propose a specialized branching scheme for the TOP. We adopt this branching scheme and restate it as follows.

Given a fractional solution $\dot{\boldsymbol{\theta}}$ of problem [P], there may exist some customers who are served for a fractional number of times and at least one route that has a fractional flow. Thus, the branching operation is performed according to two hierarchical rules.

The first rule is based on whether a customer is served or not. Specifically, we first search for a customer who is the closest to be served 0.5 times, that is, $i' = \arg\min_{i\in\mathcal{N}_c(\dot{\boldsymbol{\theta}})}\{|\sum_{r\in\mathcal{R}}\pi_{ir}\dot{\theta}_r - 0.5|\}$, where set $\mathcal{N}_c(\dot{\boldsymbol{\theta}})$ consists the customers who are served a positive number of times in the solution $\dot{\boldsymbol{\theta}}$. Then two child nodes are generated by branching on this customer. In one child node, customer i' is forced to be served. Then at the master problem level, constraint (25) for customer i' is updated to $\sum_{r\in\mathcal{R}}\pi_{i'r}\theta_r = 1$. In the other child node, customer i' is prohibited to be served. Instead of imposing $\sum_{r\in\mathcal{R}}\pi_{i'r}\theta_r = 0$ for customer i' in model [RMP], we update subset $\hat{\mathcal{R}}$ by removing routes containing customer i'. Meanwhile, in the subproblem level, we remove arcs (i', j) and (j, i') from set $\mathcal{A}$ such that any new routes found by the pricing problem will not contain customer i'. In addition, for each B&B tree node, we have a list $\mathcal{S}$ storing the customers that are constrained to be served. The list is initialized to be empty at the root node and passed from father to child nodes. When branching on a customer, we add it to the associated list if it is enforced to be served.

The second rule is based on the fractional arcs when all the selected customers are served for an integer number of times. In this case, we branch on the arc (i', j') that is the closest to be traversed 0.5 times by the routes in solution $\dot{\boldsymbol{\theta}}$, i.e., $(i', j') = \arg\min_{(i,j)\in\mathcal{A}(\dot{\boldsymbol{\theta}})}\{|\sum_{r\in\mathcal{R}}\beta_{ijr}\dot{\theta}_r - 0.5|\}$, where $\mathcal{A}(\dot{\boldsymbol{\theta}})$ is the set of arcs that are traversed by vehicles, i.e., $\mathcal{A}(\dot{\boldsymbol{\theta}}) = \{(i, j) \in \mathcal{A} | \sum_{r\in\mathcal{R}}\beta_{ijr}\dot{\theta}_r > 0\}$. Instead of a binary branching which is commonly used for fractional arcs in the VRP, we consider two cases based on whether customer i' or j' is constrained to be served. We can get this information from the list defined in the first rule. In the first case where customer i' or j' is constrained to be served, two branches are created based on arc (i', j'), where one forces the use of this arc and the other one forbids the use of this arc. In the second case where neither customer i' nor j' is constrained to be served, three branches are derived. The first branch forces to serve customer i' as well as to use arc (i', j'); the second branch forces to serve customer i', but forbids the use of arc (i', j'); and the third branch forbids to serve customer i'. When arc (i', j') is enforced to be used in a branch, arcs (i', k'), $k' \neq j'$ and (k', j'), $k' \neq i'$ are removed from set $\mathcal{A}$ for the pricing problem, and routes including these arcs are removed from $\hat{\mathcal{R}}$

for the master problem. When arc (i', j') is forbidden, it is removed from $\mathcal{A}$ and routes using this arc are removed from $\hat{\mathcal{R}}$.

5. Tabu Search Algorithm

This section presents a TS algorithm, which mainly consists of a parallel initialization procedure and a local search procedure. We first give the framework of the TS algorithm in Section 5.1, and then describe the two procedures in Sections 5.2 and 5.3, respectively.

5.1. Framework of the Tabu Search Algorithm

At the beginning of the TS algorithm, we initialize a robust feasible solution $sol = \{r_1, \ldots, r_K\}$, which has the same meaning as in Section 4.1. We then implement multiple iterations of local search. In each iteration, eight neighborhood moves are applied to the current solution sol' sequentially, which are followed by a random start procedure to escape from a local optimal solution. Finally, the algorithm terminates when the best-found solution has not been improved in $\Gamma_{\max}$ consecutive iterations. We set $\Gamma_{\max} = 50$ based on some preliminary tests.

To avoid searching the same space for several times, we consider a tabu list $\mathfrak{L}$ in each neighborhood move. Specifically, if a customer has been removed from the current route, then we forbid moving the same customer back to the same route within the following η iterations, where η represents the tabu tenure. We put the forbidden move and its tabu tenure into the tabu list $\mathfrak{L}$. The value of η associated with each forbidden move is updated according to the quality of the found solution. In particular, the initial value of η is set to 10. We reduce it by 1 if the solution is improved, otherwise we increase it by 3 when the solution is not improved within five consecutive iterations. A forbidden move is allowed to be performed if it yields a better objective value than the best-found solution.

The framework of the TS algorithm is given in Algorithm 1.

5.2. Initialization Procedure

To construct an initial feasible solution, we implement a greedy insertion method that builds K routes in parallel. First, we initialize K routes, where each route only visits the starting and the ending depots. We denote the last customer served by route r_k as ι_k and assume that an unserved customer i is inserted after ι_k. Then $\Delta t_{\iota_k i} = t_{\iota_k i} + t_{i0} - t_{\iota_k 0}$ is the incremental travel time when customer i is added to route r_k. We prioritize inserting a customer with a large profit and a small decay ratio into a route that is near this customer.

Algorithm 1 Framework of the tabu search algorithm

Input: A group of neighborhood moves $N = \{N_l | l = 1, \ldots, 8\}$ and the stop criterion $\Gamma_{\max}$.
Set the tabu list $\mathfrak{L} = \emptyset$ and the counter $\tau = 0$.
Initialize a robust feasible solution $sol = \{r_k | k = 1, \ldots, K\}$ and let $sol^* = sol$.
while $\tau < \Gamma_{\max}$ **do**
    Let $sol' = sol$;
    for $N_l, l = 1, \ldots, 8$ **do**
        $sol'' = Neighborhood\ Search(N_l, sol', \mathfrak{L})$.
        if $f(sol'') > f(sol')$ **then** $sol' = sol''$, and update $\mathfrak{L}$ **end**
    end for
    if $f(sol') > f(sol)$ **then** $sol = sol'$ **else** $sol \leftarrow RandomRestart$ **end**
    if $f(sol') > f(sol^*)$ **then** $sol^* = sol'$, $\tau = 0$ **else** $\tau = \tau + 1$ **end**
end while
Output: Best-found solution sol^*.

To do so, we define a score $\omega_{ik} = \frac{p_i}{d_i \Delta t_{\iota_k i}}$ for each combination of unserved customer $i \in \mathcal{N}_c$ and route $r_k \in sol$. If it is infeasible to insert customer i into route r_k, i.e., vehicle k cannot return to the ending depot before $T_{\max}$ or the arrival time at customer i exceeds its deadline D_i, we set $\omega_{ik} = 0$. After obtaining all the scores, we insert the customer with the highest positive score to its corresponding route. If we cannot find a score that is larger than 0, the procedure terminates. Note that we set the service time at each customer to the worst-case value, in order to guarantee the robust feasibility of the initial solution. The pseudocode of this initialization procedure is provided in Algorithm 2.

5.3. Improvement Procedure

In the local search phase, we use eight neighborhood moves, which are defined as follows:

- *Insertion:* Insert an unserved customer into the solution.
- *Removal:* Remove a served customer from the solution.
- *Replacement:* Replace a served customer i by an unserved customer j.
- *Swap:* Swap the positions of two customers in the same route.
- λ-*Exchange:* Change the locations of two chains of λ consecutive customers located in a pair of routes (r_i, r_j), where $\lambda \leq \lceil \min\{|r_i|, |r_j|\}/2 \rceil$.
- *Cross:* Remove two arcs from two routes and replace them with other two arcs.
- *Or-opt:* Relocate a chain of $\sigma \in \{1, 2, 3\}$ consecutive customers in the same route.

Algorithm 2 Initialization of a robust feasible solution

Input: The customer set $\mathcal{N}_c = \{1, \ldots, n\}$, the vehicle set $\mathcal{K} = \{1, \ldots, K\}$ and the time horizon $T_{\max}$.
Let $sol = \{r_k = \{0,0\} | \forall k \in \mathcal{K}\}$; denote the last customer of every route r_k as ι_k, and the arrival time at the ending depot as τ_k.
Initial $\iota_k \leftarrow 0$, and $\tau_k \leftarrow 0$.
repeat
  for all $i \in \mathcal{N}_c$ **do**
    for all $k \in \mathcal{K}$ **do**
      Insert i into route r_k, and place it between ι_k and the ending depot 0, then $a_{ik} = a_{\iota_k k} + \bar{s}_{\iota_k} + \hat{s}_{\iota_k} + t_{\iota_k i}$, and the time arriving at the ending depot is $\tau_k = a_{ik} + \bar{s}_i + \hat{s}_i + t_{i0}$.
      if $a_{ik} < D_i$ and $\tau_k \le T_{\max}$ **then**
        $\omega_{ik} = \frac{p_i}{d_i \Delta t_{\iota_k i}}$
      else
        $\omega_{ik} = 0$
      end if
    end for
  end for
  Let $\zeta = \max_{i \in \mathcal{N}_c, k \in \mathcal{K}} \{\omega_{ik}\}$.
  if $\zeta > 0$ **then**
    Let $(i^*, k^*) \leftarrow \arg\max_{i \in \mathcal{N}_c, k \in \mathcal{K}} \{\omega_{ik}\}$, update $\tau_{k^*} \leftarrow a_{i^* k^*} + \bar{s}_{i^*} + \hat{s}_{i^*} + t_{i^* 0}$, $\iota_{k^*} \leftarrow i^*$, $r_{k^*} = \{0, \ldots, i^*, 0\}$, $\mathcal{N}_c \leftarrow \mathcal{N}_c \setminus i^*$.
  end if
until $\zeta = 0$
Output: *sol*.

In each iteration, we implement these neighborhoods in the given sequence. For each neighborhood, we use the best improvement strategy. Finally we return the best improved neighborhood if any.

As all the eight neighborhoods are deterministic, identical solutions would be generated if the current solution is the same at the beginning of each iteration. To diversify the local search, we employ a random start to generate a new solution if the current solution is not improved in an iteration. Specifically, we generate K empty routes with only the starting and the ending depots. We then sort all customers in a descending order by profit, and assume that the service time at each customer reaches the worst-case value. Finally, we check these customers successively. For each customer, we first generate a random number ψ in the interval $[0, 1]$. If $\psi \ge 0.5$, this customer is considered for being inserted into the

solution. If there exist multiple robust feasible positions to place it, we randomly pick one for it. If $\psi < 0.5$, we move to the next customer. The insertion operations continue until all the customers have been checked.

6. Computational Study

In this section, we introduce the data set and evaluate the solution methods and the robust model. All experiments were run on a MacOS PC with a 2.7 GHz Intel Core i5 processor and 8 GB of memory using a single thread. The B&P algorithm is implemented in C++ programming language using CPLEX 12.8 as the solver. The TS method is coded in C# programming language.

6.1. Instance Sets

We generate instances based on Solomon's data sets (Solomon 1987) for the VRP with time windows. Specifically, we use Solomon's C1, R1, and RC1 data sets, where customers are clustered, randomly distributed, and a mix of clustered and randomly distributed, respectively. There are several instances in each data set with the same coordinates and demands. In our instances, the coordinates (x_i, y_i) and the demand p_i (the profit in our problem) are directly taken from Solomon's instances. The first node is treated as the depot, and the n customers are taken from the 100 customers in Solomon's data sets. We consider instances of different sizes, i.e., $n = \{25, 50, 100\}$. Instances with 25 (50) customers are generated by extracting the first 25 (50) customers from the C1 and R1 data sets. In the RC1 data set, as the first 50 customers are clustered and the second 50 customers are randomly distributed, we generate instances with 25 (50) customers by extracting the first 15 (25) customers from the first 50 customers and the rest 10 (25) customers from the second 50 customers. In each instance, the travel time t_{ij} on arc (i, j) is set to the Euclidean distance and rounded to the nearest hundredth as in Yu et al. (2022). The decay ratio d_i of customer i is a random number in the interval $[0.02, 0.03]$ and rounded to three decimal places. We assume that the larger the profit is, the longer the service time is required; thus, we set the nominal service time at customer i as $\bar{s}_i = 2p_i + 50$. We let the time horizon $T_{\max}$ take four values for each instance, which are the first, second, third, and fourth quartile of all customers' deadlines in that instance, and these values are rounded up to integers. We denote them as $T_{\max}^{Q1}$, $T_{\max}^{Q2}$, $T_{\max}^{Q3}$, and $T_{\max}^{Q4}$. The number of vehicles K takes values from the set $\{2, 3, 4, 5\}$.

For the parameters in the uncertainty set, we set the largest service time deviation of customer i as $\hat{s}_i = \epsilon \bar{s}_i$, where ϵ is a proportion belonging to the set $\{10\%, 25\%, 50\%\}$. The uncertainty budget Γ takes values from the set $\{0, 1, 5, 10\}$. Thus, there are a total of 1440 instances based on the type of data set and different values of $(\epsilon, \Gamma, n, T_{\max}, K)$, among which 1296 instances are associated with the robust model ($\Gamma > 0$) and 144 instances are for the deterministic model ($\Gamma = 0$). The details of instances and solutions are also available at `https://github.com/QinxiaoYu/RTOPDP`.

6.2. Performance Comparison Between the Two Exact Methods

This section evaluates the performance of the B&P by comparing it with CPLEX used to solve the robust model (16)–(22). Considering the limited capability of CPLEX, experiments are conducted on 25-customer instances with $T_{\max} \leq T_{\max}^{Q3}$. A computing time limit of 3600 seconds is imposed on both methods. Table 2 presents the average results grouped by the instance type and the time horizon $T_{\max}$, where the following details are provided: the number of instances #, the number of instances that are solved to optimality $\#O$ in each group, the lower bound (i.e., the best found objective value) z_{lb}, the upper bound z_{ub}, the percentage gap measured as $(z_{ub} - z_{lb})/z_{lb} \times 100$, the total computing time, the time $Time_{lb}$ that z_{lb} is reached, and the number of served customers per route $\frac{\bar{N}_c}{K}$. We note that the B&P finds optimal solutions for all the 25-customer instances; thus, only z_{lb} is reported in the table.

Table 2 Average Results of CPLEX and the B&P Algorithm for Instances with 25 Customers

			CPLEX						B&P			
Type	$T_{\max}$	#	#O	z_{lb}	z_{ub}	Gap (%)	*Time* (s)	$Time_{lb}$ (s)	z_{lb}	*Time* (s)	$Time_{lb}$ (s)	$\frac{\bar{N}_c}{K}$
C1	$T_{\max}^{Q1}$	40	0	198.32	439.73	140.2	3651.8	2945.8	213.64	9.6	5.1	2.4
	$T_{\max}^{Q2}$	40	0	265.81	442.75	80.5	3626.3	2952.0	282.22	94.4	37.0	4.8
	$T_{\max}^{Q3}$	40	0	287.73	442.36	57.2	3597.0	3419.5	289.80	48.9	33.3	5.4
	Avg.	*40*	*0*	*250.62*	*441.61*	*92.6*	*3625.0*	*3105.8*	*261.89*	*51.0*	*25.2*	*4.2*
R1	$T_{\max}^{Q1}$	40	12	129.60	227.08	71.9	2602.8	1640.6	129.95	0.5	0.4	2.0
	$T_{\max}^{Q2}$	40	0	181.70	310.91	80.2	3600.0	3362.8	183.05	6.6	4.6	3.9
	$T_{\max}^{Q3}$	40	0	183.90	311.07	76.6	3617.1	3202.1	188.09	10.1	7.5	4.5
	Avg.	*40*	*4*	*165.07*	*283.02*	*76.2*	*3273.3*	*2735.1*	*167.03*	*5.7*	*4.1*	*3.4*
RC1	$T_{\max}^{Q1}$	40	0	228.51	442.41	109.8	3600.0	3416.2	231.84	3.8	1.8	3.1
	$T_{\max}^{Q2}$	40	0	272.58	442.73	71.5	3568.7	3433.2	276.17	94.3	45.6	4.9
	$T_{\max}^{Q3}$	40	0	278.04	442.65	65.1	3600.0	3493.1	279.93	37.0	19.2	5.3
	Avg.	*40*	*0*	*259.71*	*442.60*	*82.2*	*3589.6*	*3447.5*	*262.64*	45.0	22.2	4.4
Avg.		*40*	*1*	*225.13*	*389.08*	*83.7*	*3496.0*	*3096.1*	*230.52*	*33.9*	*17.2*	*4.0*

Table 2 shows that the B&P outperforms solving the compact model using CPLEX. In particular, the B&P can solve all the 360 instances to optimality with an average

computing time of 33.9 seconds, whereas CPLEX can only solve 12 instances to optimality. The average optimality gap generated by CPLEX is 83.7% for all the instances, and it goes up to 140.2% for C1-type instances with $T_{\max}^{Q1}$. We observe that the differences between the lower bounds produced by the two methods are not significant, and the large optimality gaps of CPLEX mainly result from its weak upper bounds. In addition, we notice that no C1-type instance is solved to optimality by CPLEX when $T_{\max}$ equals to $T_{\max}^{Q3}$, but the average computing time is less than 3600 seconds. This is because CPLEX runs out-of-memory for two instances, terminating at 3060 and 3578 seconds, respectively. A similar situation appears in one RC1-type instance with $T_{\max}^{Q2}$, where CPLEX ends at 2346 seconds.

6.3. Sensitivity Analysis of the B&P Algorithm

In this section, we analyze the impacts of critical parameters on the performance of the B&P algorithm, including the time horizon $T_{\max}$ and the uncertainty-related parameters Γ and ϵ. Experiments are conducted on all the 1440 instances introduced in Section 6.1. We note that the B&P cannot provide an upper bound for 14 instances (1, 6, and 7 instances in the C1, R1, and RC1 data sets, respectively) when $|\mathcal{N}_c| = 100$ within the 3600-second time limit. After further observation of these 14 instances, we find that they are associated with long time horizons and slight derivations of service time, leading to long feasible routes. These characteristics make the pricing problem quite tricky and time-consuming such that the root node of the B&B tree cannot be solved within the time limit. Thus, the gaps between the upper and lower bounds reported in the following tables are calculated by only considering instances for which the upper bounds are generated.

6.3.1. The Impact of $T_{\max}$. Table 3 reports the average results of the B&P under different values of $T_{\max}$, where #Node is the number of nodes explored in the B&B tree and $Time_L$ is the average time consumption of each call of the labeling algorithm.

Table 3 shows that for 25-customer instances, the computing time of the B&P first increases and then decreases with $T_{\max}$. For 50-customer instances, a similar trend is observed in the number of optimally solved instances. Specifically, for each data set, all the instances can be solved to optimality when $T_{\max}$ is set to $T_{\max}^{Q1}$. Subsequently, the number of optimally solved instances decreases when the time horizon is prolonged to $T_{\max}^{Q2}$ and $T_{\max}^{Q3}$; however, this number increases when $T_{\max}$ changes from $T_{\max}^{Q3}$ to $T_{\max}^{Q4}$. From these observations, we can conclude that the impact of $T_{\max}$ on the performance of the B&P is

Table 3 Average Results of the B&P Algorithm Under Different Values of $T_{\max}$

		C1-Type Instances						R1-Type Instances						RC1-Type Instances					
$\lvert\mathcal{N}_c\rvert$	$T_{\max}$	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{N}_c}{K}$	#Node	$Time_L$ (s)	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{N}_c}{K}$	#Node	$Time_L$ (s)	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{N}_c}{K}$	#Node	$Time_L$ (s)
25	$T_{\max}^{Q1}$	40	0.00	9.6	2.4	18	0.19	40	0.00	0.5	2.0	2	0.12	40	0.00	3.8	3.1	14	0.10
	$T_{\max}^{Q2}$	40	0.00	94.4	4.8	38	0.78	40	0.00	6.6	3.9	4	0.25	40	0.00	94.3	4.9	101	0.54
	$T_{\max}^{Q3}$	40	0.00	48.9	5.4	8	0.94	40	0.00	10.1	4.5	5	0.34	40	0.00	37.0	5.3	29	0.90
	$T_{\max}^{Q4}$	40	0.00	32.4	5.4	7	0.91	40	0.00	3.7	4.6	3	0.27	40	0.00	24.5	5.5	17	0.99
	Avg.	*40*	*0.00*	*46.3*	*4.5*	*18*	*0.71*	*40*	*0.00*	*5.2*	*3.7*	*3*	*0.24*	*40*	*0.00*	*39.9*	*4.7*	*40*	*0.63*
50	$T_{\max}^{Q1}$	40	0.00	2.4	2.2	1	0.39	40	0.00	1.4	2.0	2	0.26	40	0.00	0.6	2.2	1	0.12
	$T_{\max}^{Q2}$	36	0.02	508.5	3.1	221	0.89	39	0.00	238.9	3.9	43	1.41	27	0.17	1454.2	5.0	149	5.11
	$T_{\max}^{Q3}$	30	0.03	1676.4	5.9	116	6.31	38	0.01	501.4	5.4	35	3.19	25	0.21	1854.6	6.4	122	10.72
	$T_{\max}^{Q4}$	37	0.00	981.4	6.7	58	7.90	40	0.00	304.4	6.2	13	4.77	26	0.33	1911.4	7.1	64	13.85
	Avg.	*36*	*0.01*	*792.2*	*4.5*	*99*	*3.87*	*39*	*0.00*	*261.6*	*4.4*	*23*	*2.41*	*30*	*0.18*	*1305.2*	*5.2*	*84*	*7.45*
100	$T_{\max}^{Q1}$	40	0.00	3.1	2.0	1	0.71	40	0.00	2.0	1.9	1	0.49	40	0.00	2.8	2.1	1	0.58
	$T_{\max}^{Q2}$	40	0.00	63.4	2.8	8	1.65	40	0.00	422.3	3.6	17	6.88	28	0.14	1410.5	4.3	37	18.24
	$T_{\max}^{Q3}$	13	0.79	2749.2	5.6	46	31.02	21	0.23	2432.8	5.4	21	47.37	10	1.22	3077.9	5.9	32	69.37
	$T_{\max}^{Q4}$	7	3.55	3289.9	7.5	22	52.00	10	0.97	3198.2	7.3	10	97.22	5	3.04	3371.1	7.3	13	98.40
	Avg.	*25*	*1.07*	*1526.4*	*4.5*	*19*	*21.34*	*28*	*0.27*	*1513.8*	*4.5*	*12*	*37.24*	*21*	*1.03*	*1965.6*	*4.9*	*21*	*45.99*
Avg.		*34*	*0.36*	*788.3*	*4.5*	*45*	*8.64*	*36*	*0.09*	*593.5*	*4.2*	*13*	*13.20*	*30*	*0.39*	*1103.6*	*4.9*	*48*	*17.91*

not monotonic. A longer time horizon does not necessarily slow down the B&P. We consider the reason is that although the average time of the labeling algorithm sees a slight increase, the number of explored B&B nodes reduces significantly.

For 100-customer instances, the number of optimally solved instances decreases with $T_{\max}$. Meanwhile, the average computing time increases, because a larger $T_{\max}$ allows more customers to be served by a vehicle, leading to more time consumption for the labeling process. We observe that the values of $Time_L$ for 100-customer instances are much larger than those of 25- and 50-customer instances, especially when the time horizon is set to $T_{\max}^{Q3}$ and $T_{\max}^{Q4}$. Thus, the difference in the computing time is more pronounced as the labeling algorithm is called thousands of times in the B&P.

6.3.2. The Impact of Uncertainty-Related Parameters. Tables 4 and 5 present the average results of the B&P grouped by the uncertainty budget Γ and the proportion of service time deviations ϵ, respectively. We observe that when the values of these two parameters increase, the number of optimally solved instances increases, and the computing time decreases in general. The reason for this phenomenon is that a larger value of Γ or ϵ increases the probability of deadline violations at customers; thus, the number of customers on each route becomes more limited as indicated by $\frac{\bar{N}_c}{K}$ in these tables. Meanwhile, the computing time of the labeling algorithm $Time_L$ reduces due to fewer robust feasible path extensions. In addition, we notice that the computing times of the B&P for the robust models are longer than the deterministic model in some cases, e.g., 25-customer instances of RC1-type with $\Gamma \geq 1$ and 50-customer instances of C1-type with $\epsilon = 10\%$. We consider

the increased computing time in these cases mainly stems from the increased number of explored B&B tree nodes.

Table 4 Average Results of the B&P Algorithm Under Different Values of Γ

		C1-Type Instances						R1-Type Instances						RC1-Type Instances					
$\lvert\mathcal{N}_c\rvert$	Γ	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{\mathcal{N}}_c}{K}$	#Node	$Time_L$ (s)	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{\mathcal{N}}_c}{K}$	#Node	$Time_L$ (s)	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{\mathcal{N}}_c}{K}$	#Node	$Time_L$ (s)
25	0	16	0.00	86.3	5.0	11	1.29	16	0.00	6.1	4.2	3	0.31	16	0.00	18.0	5.2	6	1.11
	1	48	0.00	62.1	4.8	29	0.82	48	0.00	6.9	3.9	5	0.26	48	0.00	26.0	5.0	21	0.78
	5	48	0.00	31.6	4.3	14	0.55	48	0.00	4.0	3.5	2	0.22	48	0.00	59.3	4.5	66	0.47
	10	48	0.00	32.0	4.3	13	0.56	48	0.00	4.4	3.5	3	0.23	48	0.00	41.7	4.5	44	0.48
	Avg.	*40*	*0.00*	*46.3*	*4.5*	*18*	*0.71*	*40*	*0.00*	*5.2*	*3.7*	*3*	*0.24*	*40*	*0.00*	*39.9*	*4.7*	*40*	*0.63*
50	0	13	0.02	831.1	5.0	31	6.30	14	0.01	898.7	4.9	95	3.59	9	0.22	2088.8	5.9	84	12.29
	1	42	0.01	967.5	4.8	92	4.94	47	0.01	351.9	4.7	24	2.99	33	0.30	1425.0	5.6	73	9.53
	5	44	0.02	751.0	4.3	116	3.01	48	0.00	102.5	4.1	11	1.84	38	0.08	1046.9	4.9	78	5.79
	10	44	0.01	645.1	4.2	112	2.86	48	0.00	117.8	4.1	12	2.00	38	0.14	1182.4	4.8	101	5.42
	Avg.	*36*	*0.01*	*792.2*	*4.5*	*99*	*3.87*	*39*	*0.00*	*261.6*	*4.4*	*23*	*2.41*	*30*	*0.18*	*1305.2*	*5.2*	*84*	*7.45*
100	0	9	2.53	1892.2	4.8	17	35.62	8	0.20	2179.2	5.1	13	69.82	5	1.01	2511.9	5.6	18	81.01
	1	25	1.50	1824.2	4.8	20	26.91	30	0.46	1804.5	4.9	13	49.13	19	1.66	2364.5	5.2	25	60.82
	5	32	0.82	1343.0	4.3	22	14.91	36	0.20	1346.2	4.3	12	26.85	29	0.75	1726.5	4.6	20	31.96
	10	34	0.38	1290.1	4.2	18	17.46	37	0.20	1169.0	4.2	11	25.37	30	0.71	1623.7	4.5	18	34.56
	Avg.	*25*	*1.07*	*1526.4*	*4.5*	*19*	*21.34*	*28*	*0.27*	*1513.8*	*4.5*	*12*	*37.24*	*21*	*1.03*	*1965.6*	*4.9*	*21*	*45.99*

Table 5 Average Results of the B&P Algorithm Under Different Values of ϵ

		C1-Type Instances						R1-Type Instances						RC1-Type Instances					
$\lvert\mathcal{N}_c\rvert$	ϵ	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{\mathcal{N}}_c}{K}$	#Node	$Time_L$ (s)	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{\mathcal{N}}_c}{K}$	#Node	$Time_L$ (s)	$\#O$	Gap (%)	$Time$ (s)	$\frac{\bar{\mathcal{N}}_c}{K}$	#Node	$Time_L$ (s)
25	0	16	0.00	86.3	5.0	11	1.29	16	0.00	6.1	4.2	3	0.31	16	0.00	18.0	5.2	6	1.11
	10%	48	0.00	61.0	4.8	26	0.81	48	0.00	7.4	3.9	5	0.27	48	0.00	48.0	5.0	33	0.83
	25%	48	0.00	41.2	4.5	16	0.65	48	0.00	4.7	3.7	3	0.24	48	0.00	44.7	4.7	55	0.56
	50%	48	0.00	23.5	4.0	13	0.47	48	0.00	3.2	3.3	2	0.20	48	0.00	34.2	4.3	43	0.35
	Avg.	*40*	*0.00*	*46.3*	*4.5*	*18*	*0.71*	*40*	*0.00*	*5.2*	*3.7*	*3*	*0.24*	*40*	*0.00*	*39.9*	*4.7*	*40*	*0.63*
50	0	13	0.02	831.1	5.0	31	6.30	14	0.01	898.7	4.9	95	3.59	9	0.22	2088.8	5.9	84	12.29
	10%	42	0.01	990.7	4.7	93	5.05	47	0.01	315.4	4.7	24	3.10	31	0.31	1582.8	5.5	79	10.02
	25%	42	0.02	758.6	4.5	146	3.35	48	0.00	154.8	4.4	14	2.22	39	0.07	1140.6	5.2	75	6.47
	50%	46	0.01	614.4	4.0	80	2.41	48	0.00	102.1	3.9	8	1.51	39	0.15	931.0	4.6	98	4.25
	Avg.	*36*	*0.01*	*792.2*	*4.5*	*99*	*3.87*	*39*	*0.00*	*261.6*	*4.4*	*23*	*2.41*	*30*	*0.18*	*1305.2*	*5.2*	*84*	*7.45*
100	0	9	2.53	1892.2	4.8	17	35.62	8	0.20	2179.2	5.1	13	69.82	5	1.01	2511.9	5.6	18	81.01
	10%	24	1.72	1893.8	4.7	21	28.90	29	0.41	1790.6	4.9	10	46.15	19	1.78	2354.6	5.1	23	58.43
	25%	28	0.66	1598.5	4.5	23	18.43	33	0.15	1519.0	4.5	16	34.02	25	1.02	1973.2	4.8	21	42.25
	50%	39	0.34	964.9	4.1	15	11.95	41	0.29	1010.1	4.0	10	21.05	34	0.34	1386.9	4.4	20	26.61
	Avg.	*25*	*1.07*	*1526.4*	*4.5*	*19*	*21.34*	*28*	*0.27*	*1513.8*	*4.5*	*12*	*37.24*	*21*	*1.03*	*1965.6*	*4.9*	*21*	*45.99*

Based on the results in Tables 3–5, we further observe that R1-type instances are relatively more manageable than the other two types of instances. Specifically, when $|\mathcal{N}_c| = 25$, although all the instances in each data set can be solved to optimality, the average computing time of the R1-type instances is the shortest. When $|\mathcal{N}_c| = 50$ and 100, more R1-type instances can be optimally solved. We consider the reason is that customers in the R1 data set are randomly distributed, and then fewer can be included in a vehicle route as suggested by the indicator $\frac{\mathcal{N}_c}{K}$.

6.4. Performance Comparison Between the B&P and the TS Algorithms

This section compares the performance between the B&P and the TS algorithms. For the TS, we run it ten times for each instance. Table 6 reports the average results, where column $\#B$ refers to the number of instances that the B&P can provide an upper bound within the time limit, based on which the average values of other performance indicators are reported in each row. For the TS algorithm, we provide the best, average, and worst objective values, the relative standard deviation (RSD) among ten runs, the percentage gap Gap_t between z_{ub} and $BestObj$ (i.e., $\text{Gap}_t = (z_{ub} - BestObj)/BestObj \times 100$), and the total computing time of ten runs.

Table 6 Average Results of the B&P and the TS Algorithms

		B&P							TS					
$\lvert\mathcal{N}_c\rvert$	$T_{\max}$	$\#B$	$\#O$	z_{lb}	z_{ub}	Gap (%)	$Time$ (s)	$Time_{lb}$ (s)	Best Obj	Avg. Obj	Worst Obj	RSD (%)	Gap_t (%)	$Time$ (s)
25	$T_{\max}^{Q1}$	120	120	191.81	191.81	0.00	4.6	2.4	191.59	190.88	189.68	0.31	0.08	2.9
	$T_{\max}^{Q2}$	120	120	247.14	247.14	0.00	65.1	29.1	247.11	246.84	246.40	0.10	0.01	9.9
	$T_{\max}^{Q3}$	120	120	252.61	252.61	0.00	32.0	20.0	252.58	252.40	252.14	0.07	0.01	12.4
	$T_{\max}^{Q4}$	120	120	252.96	252.96	0.00	20.2	13.9	252.93	252.77	252.48	0.06	0.01	14.4
	Avg.	*120*	*120*	*236.13*	*236.13*	*0.00*	*30.5*	*16.4*	*236.05*	*235.72*	*235.18*	*0.14*	*0.03*	*9.9*
50	$T_{\max}^{Q1}$	120	120	200.38	200.38	0.00	1.5	1.2	200.31	199.65	198.68	0.31	0.03	3.3
	$T_{\max}^{Q2}$	120	102	303.42	303.71	0.07	733.9	190.3	302.69	301.17	299.31	0.35	0.26	12.1
	$T_{\max}^{Q3}$	120	93	351.17	351.42	0.08	1344.1	534.7	350.97	350.23	349.34	0.18	0.13	29.7
	$T_{\max}^{Q4}$	120	103	356.14	356.49	0.11	1065.7	628.3	356.29	355.81	355.20	0.10	0.05	45.6
	Avg.	*120*	*104.5*	*302.78*	*303.00*	*0.07*	*786.3*	*338.6*	*302.57*	*301.72*	*300.63*	*0.23*	*0.12*	*22.7*
100	$T_{\max}^{Q1}$	120	120	212.92	212.92	0.00	2.6	2.6	212.28	211.45	210.27	0.39	0.44	4.8
	$T_{\max}^{Q2}$	120	108	335.72	335.94	0.05	632.1	248.9	334.70	332.59	329.96	0.45	0.31	14.9
	$T_{\max}^{Q3}$	118	44	437.82	441.51	0.74	2738.9	1395.6	438.74	436.76	434.53	0.33	0.56	43.2
	$T_{\max}^{Q4}$	108	22	460.48	471.37	2.57	3251.5	2168.0	469.66	468.77	467.82	0.12	0.38	108.5
	Avg.	*116.5*	*73.5*	*358.87*	*362.38*	*0.80*	*1610.6*	*920.6*	*360.80*	*359.34*	*357.57*	*0.33*	*0.42*	*41.2*
	Avg.	*118.8*	*99.3*	*298.67*	*299.90*	*0.28*	*801.3*	*420.34*	*299.21*	*298.33*	*297.21*	*0.23*	*0.19*	*24.4*

From Table 6, we observe that for instances with 50 customers, the B&P can solve 418 out of 480 instances to optimality with an average time of 786.3 seconds (about 14 minutes). For instances with 100 customers, 294 instances are optimally solved with an average time of 1610.6 seconds (about 27 minutes). Moreover, the average optimality gaps are quite small—0.07% for instances with 50 customers and 0.80% for instances with 100 customers. In addition, the results in column $Time_{lb}$ suggest that the B&P can find optimal solutions in a short time and that most time is consumed to prove the optimality of solutions. To conclude, our B&P algorithm can provide optimal or near-optimal solutions for the RTOP-DP instances in an acceptable time frame.

With respect to the TS algorithm, it finds solutions that are very close to those generated by the B&P; however, the computing time of the TS is considerably shorter. Specifically, the average gap between the z_{ub} of the B&P and the *Best Obj* of the TS is 0.19%. And ten runs of TS only consume 24.4 seconds on average, which is about 3.0% of the computing time of the B&P (801.3 seconds). We further notice that the TS is also robust as the RSD is only 0.23% on average for all the instances. In addition, we observe that, in general, both approaches require a longer computing time when the number of customers increases; however, the increased values of the TS are much smaller than those of the B&P. We also notice that the TS provides better solutions than the B&P for instances with 100 customers and a long time horizon (i.e., $T_{\max}^{Q3}$ and $T_{\max}^{Q4}$). Thus, we can use the TS algorithm to solve large-size instances of the RTOP-DP.

6.5. Robustness Analysis

In this section, we examine the quality of robust solutions via simulation tests. To this end, for each instance we generate additional 10000 simulation samples based on the same parameter setting to simulate the random realization of service times, where $\tilde{s}_i, i \in \mathcal{N}_c$ follows a uniform distribution in the interval $[\bar{s}_i, (1+\epsilon)\bar{s}_i]$. In particular, each time after obtaining the vehicle scheduling of one instance, we test its performance in the corresponding 10000 simulation samples, which is evaluated by two indicators: the price of robustness (PoR) and the risk ($Risk$). The PoR is defined as the compromise of profit for being robust (Bertsimas and Sim 2004), which is computed as $PoR = \frac{z(0)-z(\epsilon,\Gamma)}{z(0)} \times 100$, where $z(0)$ and $z(\epsilon,\Gamma)$ represent the optimal values of the deterministic and the robust models, respectively. The $Risk$ refers to the probability of deadline violation, which is measured as the ratio of the number of times that a given solution is infeasible to the 10000 simulations (Munari et al. 2019). We report the PoR, $Risk$, and the number of served customers per route $\frac{\bar{N_c}}{K}$ under different values of Γ and ϵ in Table 7, where each row provides the average results of the three types of instances with different numbers of vehicles ($K = 2, 3, 4, 5$). The rows with $\Gamma = 0$ are the results of the deterministic model; thus, indicator PoR is not applicable, which is marked as "–".

From Table 7, we get the following observations: (1) In most cases the robust solutions can significantly reduce the probability of deadline violation in simulation tests with only a slight compromise of profit, compared to the deterministic solutions. Specifically, when $\epsilon = 10\%$, the PoR is under 7.57% for all the cases while the $Risk$ is reduced nearly to

Table 7 *Risk* **and** *PoR*

			$\epsilon = 10\%$			$\epsilon = 25\%$			$\epsilon = 50\%$		
$\lvert N_c\rvert$	$T_{\max}$	Γ	*PoR* (%)	*Risk* (%)	$\frac{\bar{\mathcal{N}}_c}{K}$	*PoR* (%)	*Risk* (%)	$\frac{\bar{\mathcal{N}}_c}{K}$	*PoR* (%)	*Risk* (%)	$\frac{\bar{\mathcal{N}}_c}{K}$
25	$T_{\max}^{Q1}$	0	–	53.95	3.0	–	68.70	3.0	–	88.02	3.0
		1	3.02	16.86	2.7	4.97	30.50	2.7	11.98	36.62	2.5
		5	4.92	0.00	2.7	11.96	0.00	2.4	28.16	0.00	2.0
		10	4.92	0.00	2.7	11.96	0.00	2.4	28.16	0.00	2.0
	$T_{\max}^{Q2}$	0	–	32.58	5.2	–	53.77	5.2	–	64.72	5.2
		1	1.59	24.32	5.1	3.38	31.89	4.9	6.75	46.97	4.6
		5	3.75	0.00	4.8	8.31	0.00	4.4	15.59	0.00	3.6
		10	3.78	0.00	4.8	8.31	0.00	4.4	15.59	0.00	3.6
	$T_{\max}^{Q3}$	0	–	3.09	5.4	–	13.70	5.5	–	23.55	5.5
		1	1.19	3.01	5.4	3.00	13.69	5.4	5.94	17.00	5.1
		5	2.93	0.01	5.3	6.90	0.00	4.9	12.84	0.00	4.3
		10	3.04	0.00	5.2	6.91	0.00	4.9	12.89	0.00	4.3
	$T_{\max}^{Q4}$	0	–	0.00	5.6	–	3.40	5.6	–	6.40	5.6
		1	1.21	0.00	5.5	2.99	3.40	5.5	5.88	1.34	5.2
		5	2.94	0.00	5.4	6.90	0.00	5.0	12.56	0.00	4.6
		10	2.96	0.00	5.4	6.95	0.00	5.0	12.60	0.00	4.6
50	$T_{\max}^{Q1}$	0	–	47.44	2.4	–	73.05	2.4	–	91.53	2.4
		1	2.35	15.52	2.3	7.27	21.05	2.2	16.07	26.15	2.0
		5	6.15	0.00	2.2	13.90	0.00	2.0	32.30	0.00	1.9
		10	6.15	0.00	2.2	13.90	0.00	2.0	32.30	0.00	1.9
	$T_{\max}^{Q2}$	0	–	64.13	4.7	–	95.30	4.7	–	99.54	4.7
		1	1.91	36.89	4.5	4.51	66.12	4.4	9.78	82.96	4.2
		5	5.54	0.00	4.3	14.15	0.00	3.8	24.83	0.00	3.1
		10	5.78	0.00	4.3	14.15	0.00	3.8	24.84	0.00	3.1
	$T_{\max}^{Q3}$	0	–	35.28	6.7	–	58.51	6.7	–	75.53	6.7
		1	1.35	28.81	6.6	3.28	58.48	6.6	6.97	69.11	6.2
		5	4.22	0.09	6.3	9.91	0.04	5.7	15.34	0.00	4.8
		10	4.43	0.00	6.2	10.21	0.00	5.5	18.85	0.00	4.6
	$T_{\max}^{Q4}$	0	–	0.00	7.3	–	0.00	7.3	–	4.61	7.3
		1	1.49	0.00	7.2	3.32	0.00	7.2	6.37	0.00	6.9
		5	3.89	0.00	6.9	9.14	0.00	6.5	16.65	0.00	5.8
		10	4.21	0.00	6.8	9.39	0.00	6.4	16.72	0.00	5.7
100	$T_{\max}^{Q1}$	0	–	40.65	2.1	–	69.32	2.1	–	91.55	2.1
		1	3.07	7.24	2.0	6.66	20.19	2.0	20.17	29.48	2.0
		5	4.51	0.00	2.0	15.90	0.00	2.0	34.53	0.00	1.8
		10	4.51	0.00	2.0	15.90	0.00	2.0	34.53	0.00	1.8
	$T_{\max}^{Q2}$	0	–	85.68	4.1	–	98.84	4.1	–	99.91	4.1
		1	2.84	61.09	4.0	6.56	70.01	3.9	13.68	79.42	3.7
		5	7.57	0.00	3.9	17.77	0.00	3.3	28.29	0.00	2.7
		10	7.55	0.00	3.9	17.77	0.00	3.3	28.29	0.00	2.7
	$T_{\max}^{Q3}$	0	–	83.55	6.6	–	91.79	6.6	–	97.71	6.6
		1	1.56	49.99	6.4	3.66	83.98	6.3	7.09	95.72	6.0
		5	4.85	0.03	6.0	15.12	0.00	5.2	25.92	0.00	4.2
		10	5.13	0.00	6.0	12.74	0.00	5.2	23.32	0.00	4.3
	$T_{\max}^{Q4}$	0	–	0.00	7.7	–	4.32	7.7	–	10.81	7.7
		1	1.01	0.00	7.7	2.53	0.00	7.8	5.19	0.04	7.7
		5	4.56	0.00	7.5	8.09	0.01	7.5	16.16	0.00	6.6
		10	3.73	0.00	7.5	10.95	0.00	7.1	18.69	0.00	6.5

0 when $\Gamma = 5$ and 10. In contrast, the *Risk* of the deterministic solutions is over 30% for most cases, and sometimes it is up to 85.68%. When $\epsilon = 25\%$ and 50%, although the *PoR* increases, the *Risk* is reduced to 0 at $\Gamma = 5$. Whereas the *Risk* of the deterministic solutions increases significantly, notably, when $\epsilon = 50\%$, it is over 60% for most cases. (2) The *Risk* of robust solutions is closely related to the uncertainty budget Γ and the number

of customers in the route $\frac{\bar{N}_c}{K}$. In particular, the *Risk* reduces to 0 when Γ is larger than or equal to $\frac{\bar{N}_c}{K}$; otherwise, the larger the difference between Γ and $\frac{\bar{N}_c}{K}$ is, the higher the *Risk* is in general. Moreover, we notice that when the time horizon $T_{\max}$ is getting closer to the latest deadline of all customers (i.e., increasing from $T_{\max}^{Q3}$ to $T_{\max}^{Q4}$), the *Risk* also decreases even though the gap between $\frac{\bar{N}_c}{K}$ and Γ is large. The reason is that the time horizon is not restricted in these cases, and then the probability of deadline violation reduces. We can see that the deterministic solutions can still work well in these situations. (3) The *PoR* shows a concavely increasing trend with Γ, i.e., solutions that are more robust against deadline violations can be obtained with an increasing *PoR*; however, the amount of marginal *PoR* decreases for each unit increase of Γ. Therefore, the uncertainty budget Γ controls the trade-off between the *PoR* and the *Risk*.

As an example, Figure 1 intuitively shows the trade-off between the *PoR* and the *Risk* using a C1-type instance with $|\mathcal{N}_c| = 25, K = 2, T_{\max}^{Q2} = 667$, and $\epsilon = 50\%$, which is randomly chosen. It displays that the optimal deterministic solution (when $\Gamma = 0$) is infeasible for all the simulation instances, resulting in a 100% risk of deadline violation. With respect to the robust model, when Γ increases, the *PoR* increases and the *Risk* decreases as expected. When $\Gamma \geq 3$, the robust solution can always guarantee the feasibility of simulation

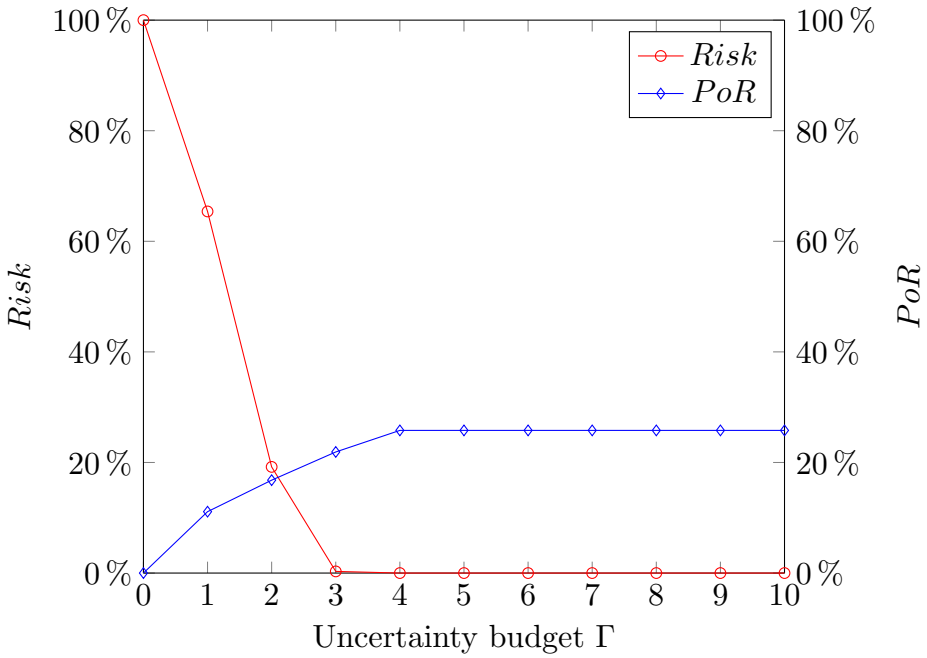

Figure 1 **(Color online) Trade-off between the** *PoR* **and the** *Risk* **for a C1-type Instance with** $|\mathcal{N}_c| = 25$, $K = 2$, $T_{\max}^{Q2} = 667$, **and** $\epsilon = 50\%$

instances, i.e., the risk of deadline violation is 0. Moreover, we can see that when Γ increases from 0 to 2, the *Risk* decreases by 80.8% and the *PoR* increases by 16.8%, which further confirms our observation that the robust model can effectively reduce the risk of deadline violation without significantly compromising the profit.

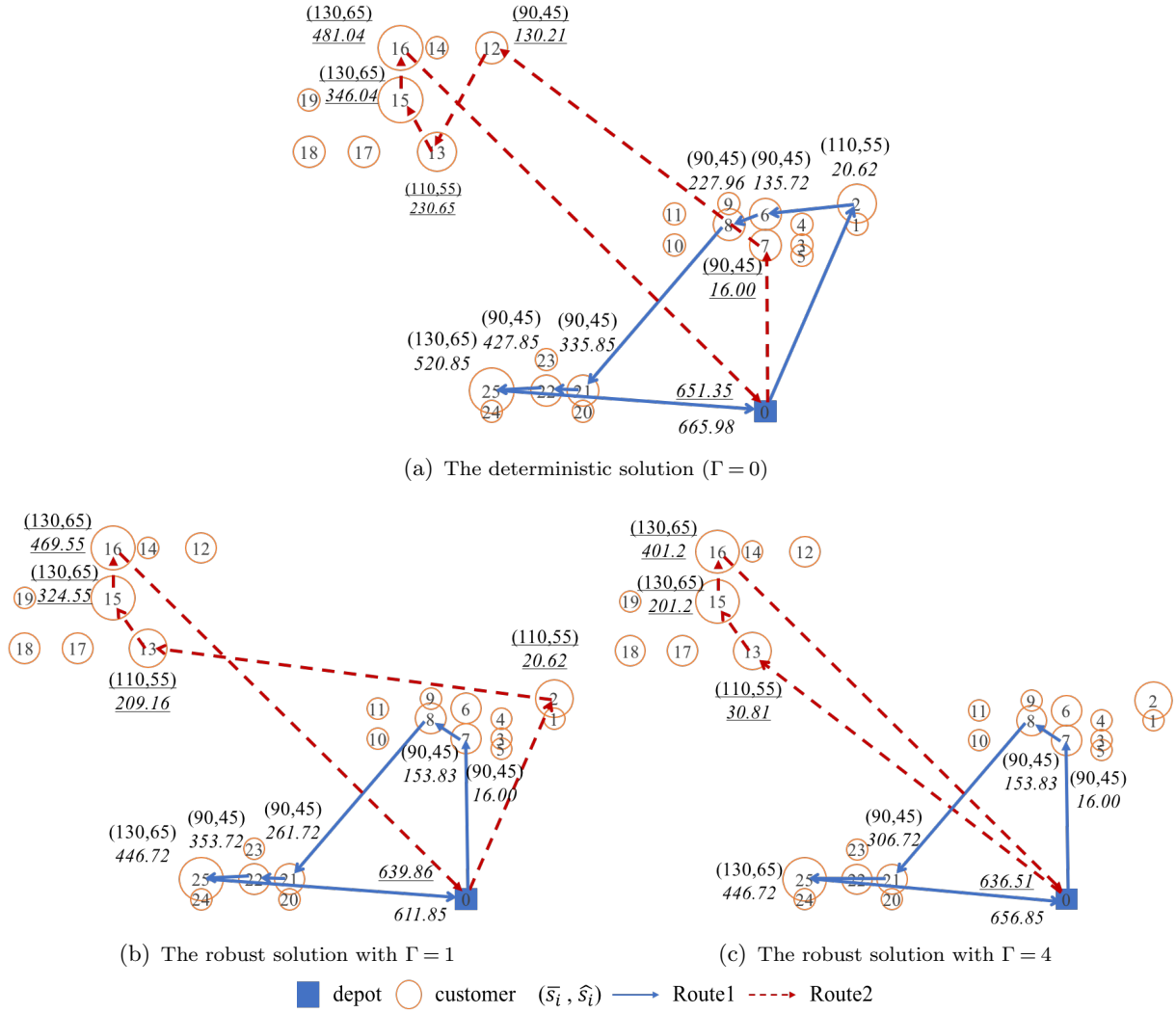

Figure 2 **(Color online) Detailed Solutions of a C1-type Instance with $|\mathcal{N}_c| = 25$, $K = 2$, $T_{\max}^{Q2} = 667$, and $\epsilon = 50\%$ Under Different Values of Γ**

We further explore the characteristics of routes in the optimal solutions under different values of Γ using the same instance, which are shown in Figure 2, where each customer is represented by a circle, whose size indicates the magnitude of profit. The vehicle routes, together with the nominal service time and its largest deviation (the values in the parenthesis) and the worst-case arrival time at each served customer, are presented. To distinguish

the route information of the two vehicles, we underline the related values of the second route.

The optimal vehicle scheduling under the deterministic model is depicted in Figure 2(a), which has a total collected profit of 236.40 with a risk of 100%. The first and second routes serve 6 and 5 customers, respectively. We notice that the difference between the total time of a route (665.98 for the first route, and 651.35 for the second one) and the time horizon (667) is smaller than all the served customers' largest deviations of service times; therefore, if any visited customer assumes its worst-case service time, the deterministic solution will violate the deadline at the ending depot and thus become infeasible.

The optimal vehicle scheduling under the robust model with $\Gamma = 1$ is shown in Figure 2(b), which has a total profit of 210.12 with a $Risk$ of 65.43% and a PoR of 11.1%. We can see that the numbers of customers served by the two routes are both reduced by one, compared to the deterministic case; and two customers (customers 2 and 7) switch to each other's route, making the served customers of Route 1 more clustered. These two changes leave more buffer times for each route to immunize against uncertainties.

The optimal solution for $\Gamma = 4$ is given in Figure 2(c), which has a total profit of 175.46 with a $Risk$ of 0% and a PoR of 25.78%. It can hedge against all the uncertainty realizations within the uncertainty set, where four customers may attain their worst-case service times, because the number of served customers on each route is not larger than four. Thus, setting $\Gamma = 4$ is sufficient for a decision maker to have a completely robust decision, and it makes no sense to set a larger value for Γ.

7. Conclusions

In this paper, we study a team orienteering problem with decreasing profits under the setting that the service times at customers are not precisely known at the time of making system decisions. We represent the uncertain service times as random variables taking values in a budgeted uncertainty set and adopt a robust optimization method to solve the problem. We aim to determine the set of served customers and the vehicle routes that maximize the total collected profit while also remaining feasible for all the uncertainty realizations within the uncertainty set. We first present a two-index vehicle flow model using constraints based on dynamic programming recursive equations and then a route-based model for the problem. A B&P algorithm is subsequently developed to solve the route-based

model for exact solutions, and a TS algorithm is implemented for approximation solutions. Extensive numerical tests are conducted to evaluate the developed solution methods and the proposed robust model. Results show that our B&P outperforms solving the compact two-index model using CPLEX, and it can solve most large-size instances to optimality in an acceptable time frame. Moreover, the TS algorithm can find high-quality solutions within a few seconds, demonstrating the effectiveness of our solution methods. Results further demonstrate that the robust model can effectively reduce the risk of deadline violation in simulation tests without significantly compromising the profit, compared to the deterministic model.

Future research can be conducted in the following aspects: (1) To produce high-quality solutions for large-size RTOP-DP instances more efficiently, we can consider adding valid inequalities to enhance the B&P method or combining it with heuristic approaches as in Alvarez and Munari (2017) and Moreno et al. (2020); (2) Other robust optimization frameworks with different types of uncertainty or ambiguity sets, e.g., the distributionally robust optimization method with a Wasserstein ambiguity set (Saif and Delage 2021) or an event-wise ambiguity set (Chen et al. 2020), can be considered for the TOP-DP with uncertainty.

Acknowledgements

This research was supported by the Fundamental Research Funds for the Central Universities, Civil Aviation University of China [Grant 3122022093] and the National Natural Science Foundation of China [Grants 72101049, 72122015, 71971154]. The authors are thankful to the associate editor and two anonymous referees for their helpful comments.

References

Afsar H, Labadie N (2013) Team orienteering problem with decreasing profits. *Electronic Notes in Discrete Mathematics* 41:285–293.

Agra A, Christiansen M, Figueiredo R, Hvattum LM, Poss M, Requejo C (2013) The robust vehicle routing problem with time windows. *Computers & Operations Research* 40(3):856–866.

Alvarez A, Munari P (2017) An exact hybrid method for the vehicle routing problem with time windows and multiple deliverymen. *Computers & Operations Research* 83:1–12.

Angelelli E, Archetti C, Filippi C, Vindigni M (2017) The probabilistic orienteering problem. *Computers & Operations Research* 81:269–281.

Balcik B, Yanıkoğlu İ (2020) A robust optimization approach for humanitarian needs assessment planning under travel time uncertainty. *European Journal of Operational Research* 282(1):40–57.

Bertsimas D, Sim M (2004) The price of robustness. *Operations Research* 52(1):35–53.

Bian Z, Liu X (2018) A real-time adjustment strategy for the operational level stochastic orienteering problem: A simulation-aided optimization approach. *Transportation Research Part E: Logistics and Transportation Review* 115:246–266.

Boussier S, Feillet D, Gendreau M (2007) An exact algorithm for team orienteering problems. *4OR* 5(3):211–230.

Campbell A, Gendreau M, Thomas B (2011) The orienteering problem with stochastic travel and service times. *Annals of Operations Research* 186(1):61–81.

Chao I, Golden B, Wasil E (1996) The team orienteering problem. *European Journal of Operational Research* 88(3):464–474.

Chen Z, Sim M, Xiong P (2020) Robust stochastic optimization made easy with rsome. *Management Science* 66(8):3329–3339.

Costa L, Contardo C, Desaulniers G (2019) Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science* 53(4):946–985.

Desrochers M, Desrosiers J, Solomon M (1992) A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research* 40(2):342–354.

Dewilde T, Cattrysse D, Coene S, Spieksma FCR, Vansteenwegen P (2013) Heuristics for the traveling repairman problem with profits. *Computers & Operations Research* 40(7):1700–1707.

Dolinskaya I, Shi ZE, Smilowitz K (2018) Adaptive orienteering problem with stochastic travel times. *Transportation Research Part E: Logistics and Transportation Review* 109:1–19.

Dror M (1994) Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research* 42(5):977–978.

Ekici A, Retharekar A (2013) Multiple agents maximum collection problem with time dependent rewards. *Computers & Industrial Engineering* 64(4):1009–1018.

Evers L, Barros AI, Monsuur H, Wagelmans A (2014a) Online stochastic UAV mission planning with time windows and time-sensitive targets. *European Journal of Operational Research* 238(1):348–362.

Evers L, Dollevoet T, Barros AI, Monsuur H (2014b) Robust UAV mission planning. *Annals of Operations Research* 222(1):293–315.

Evers L, Glorie K, van der Ster S, Barros AI, Monsuur H (2014c) A two-stage approach to the orienteering problem with stochastic weights. *Computers & Operations Research* 43:248–260.

Feillet D (2010) A tutorial on column generation and branch-and-price for vehicle routing problems. *4OR* 8(4):407–424.

Houck D, Picard J, Queyranne M, Vemuganti R (1980) The travelling salesman problem as a constrained shortest path problem: Theory and computational experience. *Opsearch* 17:93–109.

İlhan T, Iravani SMR, Daskin MS (2008) The orienteering problem with stochastic profits. *IIE Transactions* 40(4):406–421.

INSARAG (2020) *INSARAG GUIDELINES 2020 – Volume II: Preparedness and Response*. The International Search and Rescue Advisory Group, URL `https://www.insarag.org/wp-content/uploads/2021/06/INSARAG20Guidelines20Vol20II2C20Man20B.pdf`.

Irnich S, Villeneuve D (2006) The shortest-path problem with resource constraints and k–cycle elimination for $k \geq 3$. *INFORMS Journal on Computing* 18(3):391–406.

Jesper L (1999) *Parallelization of the vehicle routing problem with time windows*. Ph.D. thesis, IMM - DTU Technical University of Denmark, Kgs. Lyngby.

Jin H, Thomas BW (2019) Team orienteering with uncertain rewards and service times with an application to phlebotomist intrahospital routing. *Networks* 73(4):453–465.

Kohl N (1995) *Exact methods for time constrained routing and related scheduling problems*. Ph.D. thesis, Technical University of Denmark, Kgs. Lyngby.

Liao Z, Zheng W (2018) Using a heuristic algorithm to design a personalized day tour route in a time-dependent stochastic environment. *Tourism Management* 68:284–300.

Miller CE, Tucker AW, Zemlin RA (1960) Integer programming formulation of traveling salesman problems. *Journal of the ACM* 7(4):326–329.

Moreno A, Munari P, Alem D (2020) Decomposition-based algorithms for the crew scheduling and routing problem in road restoration. *Computers & Operations Research* 119:104935.

Munari P, Moreno A, De La Vega J, Alem D, Gondzio J, Morabito R (2019) The robust vehicle routing problem with time windows: Compact formulation and branch-price-and-cut method. *Transportation Science* 53(4):1043–1066.

Papapanagiotou V, Montemanni R, Gambardella LM (2016) A sampling-based metaheuristic for the orienteering problem with stochastic travel times. *Theory and Practice of Natural Computing*, 97–109, TPNC 2016 (Cham: Springer).

Saif A, Delage E (2021) Data-driven distributionally robust capacitated facility location problem. *European Journal of Operational Research* 291(3):995–1007.

Santini A, Plum CEM, Ropke S (2018) A branch-and-price approach to the feeder network design problem. *European Journal of Operational Research* 264(2):607–622, ISSN 0377-2217.

Smith JE, Winkler RL (2006) The optimizer's curse: Skepticism and postdecision surprise in decision analysis. *Management Science* 52(3):311–322.

Solomon MM (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research* 35(2):254–265.

Song Y, Ulmer MW, Thomas BW, Wallace SW (2020) Building trust in home services–stochastic team-orienteering with consistency constraints. *Transportation Science* 54(3):823–838.

Tang H, Miller-Hooks E (2005) Algorithms for a stochastic selective travelling salesperson problem. *Journal of the Operational Research Society* 56(4):439–452.

Tang H, Miller-Hooks E, Tomastik R (2007) Scheduling technicians for planned maintenance of geographically distributed equipment. *Transportation Research Part E: Logistics and Transportation Review* 43(5):591–609.

Varakantham P, Kumar A, Lau HC, Yeoh W (2018) Risk-sensitive stochastic orienteering problems for trip optimization in urban environments. *ACM Transactions on Intelligent Systems and Technology* 9(3):1–25.

Verbeeck C, Vansteenwegen P, Aghezzaf EH (2016) Solving the stochastic time-dependent orienteering problem with time windows. *European Journal of Operational Research* 255(3):699–718.

Yu Q, Adulyasak Y, Rousseau LM, Zhu N, Ma S (2022) Team orienteering with time-varying profit. *INFORMS Journal on Computing* 34(1):262–280.

Zhang S, Ohlmann JW, Thomas BW (2014) A priori orienteering with time windows and stochastic wait times at customers. *European Journal of Operational Research* 239(1):70–79.

Zhang S, Ohlmann JW, Thomas BW (2018) Dynamic orienteering on a network of queues. *Transportation Science* 52(3):691–706.

Zhang S, Ohlmann JW, Thomas BW (2020) Multi-period orienteering with uncertain adoption likelihood and waiting at customers. *European Journal of Operational Research* 282(1):288–303.

Online Supplement
Robust Team Orienteering Problem with Decreasing Profit

Qinxiao Yu[1], Chun Cheng*[2], and Ning Zhu[3]

[1]Economics and Management College, Civil Aviation University of China, Tianjin 300300, China
[2]Institute of Supply Chain Analytics, Dongbei University of Finance and Economics, Dalian 116025, China
*Corresponding author
[3]Institute of Systems Engineering, College of Management and Economics, Tianjin University, Tianjin 300072, China

Appendix A Extension to Travel Time Uncertainty

Besides uncertain service times, our modeling framework can also be extended to include uncertain travel times. This section discusses the modifications to the uncertainty set, the recursive equations, and the robust model for accommodating this extension.

A.1 Uncertainty Set for Travel Times

As the uncertain service times, we also assume that the uncertain travel times are independent random variables falling within a range of estimated values denoted as

$$\tilde{t}_{ij} = \bar{t}_{ij} + \chi_{ij}\hat{t}_{ij}, \chi_{ij} \in [-1, 1], \forall (i,j) \in \mathcal{A}, \tag{A.1}$$

where $\bar{t}_{ij}$ is the nominal travel time between nodes $i \in \mathcal{N}$ and $j \in \mathcal{N}$; $\hat{t}_{ij}$ is the largest deviation of travel time from its nominal value; and χ_{ij} is the uncertain parameter belonging to $[-1, 1]$. Similar to the uncertainty set $\mathcal{U}$ for the service times, we also employ a budgeted uncertainty set $\mathcal{V}$ for the travel times, which is written as

$$\mathcal{V} = \left\{ \tilde{\boldsymbol{t}} \in \mathbb{R}^{|\mathcal{A}|} : \tilde{t}_{ij} = \bar{t}_{ij} + \chi_{ij}\hat{t}_{ij}, \sum_{(i,j)\in\mathcal{A}} \chi_{ij} \le \Upsilon,\ 0 \le \chi_{ij} \le 1, \forall (i,j) \in \mathcal{A} \right\}, \tag{A.2}$$

where parameter Υ is the *uncertainty budget*, which takes an integer value in the interval $[0, |\mathcal{A}|]$. Υ controls the level of conservatism by restricting the number of arcs whose travel times are allowed to deviate from their nominal values.

A.2 Modification to the Recursive Equations

Recall that $r = (v_0, v_1, v_2, \ldots, v_m, v_{m+1})$ represents a route that starts from the depot $v_0 = 0$, sequentially serves m customers, and finally returns to the depot $v_{m+1} = 0$. To calculate the worst-case arrival time at each node, we need to consider uncertainties stemming from both the travel times and the service times. In this case, another index for travel time uncertainty is necessary, thus we modify the original decision variable $a_{i\gamma}$ to $a_{i\gamma\varsigma}$, which denotes the arrival time at node $i \in \mathcal{N}$ when $\gamma \in \{0, 1, \ldots, \Gamma\}$ predecessors' service times and $\varsigma \in \{0, 1, \ldots, \Upsilon\}$ traversed arcs' travel times have reached their worst-case values. To compute $a_{i\gamma\varsigma}$, we can still use the recursive equations provided in Section 3.3.2 with some modifications. For simplicity of notation, we use index j to represent node v_j in route r, i.e., we replace $a_{v_j\gamma\varsigma}$ by $a_{j\gamma\varsigma}$. We then calculate $a_{j\gamma\varsigma}, \forall j \in \{0, 1, \ldots, m+1\}$ as follows:

$$
a_{j\gamma\varsigma} = \begin{cases}
0, & j = 0,\ 0 \le \gamma \le \Gamma,\ 0 \le \varsigma \le \Upsilon, \quad \text{(A.3a)} \\
\bar{t}_{0j}, & j = 1,\ 0 \le \gamma \le \Gamma,\ \varsigma = 0, \quad \text{(A.3b)} \\
\bar{t}_{0j} + \hat{t}_{0j}, & j = 1,\ 0 \le \gamma \le \Gamma,\ 0 < \varsigma \le \Upsilon, \quad \text{(A.3c)} \\
a_{i\gamma\varsigma} + \bar{s}_i + \bar{t}_{ij}, & j > 1,\ i = j-1,\ \gamma = 0,\ \varsigma = 0, \quad \text{(A.3d)} \\
\max\Big\{ a_{i\gamma\varsigma} + \bar{s}_i + \bar{t}_{ij}, & \\
\quad a_{i\gamma-1\varsigma} + \bar{s}_i + \hat{s}_i + \bar{t}_{ij} \Big\} & j > 1,\ i = j-1,\ 0 < \gamma \le \Gamma,\ \varsigma = 0, \quad \text{(A.3e)} \\
\max\Big\{ a_{i\gamma\varsigma} + \bar{s}_i + \bar{t}_{ij}, & \\
\quad a_{i\gamma\varsigma-1} + \bar{s}_i + \bar{t}_{ij} + \hat{t}_{ij} \Big\} & j > 1,\ i = j-1,\ 0 < \varsigma \le \Upsilon,\ \gamma = 0, \quad \text{(A.3f)} \\
\max\Big\{ a_{i\gamma\varsigma} + \bar{s}_i + \bar{t}_{ij}, & \\
\quad a_{i\gamma-1\varsigma} + \bar{s}_i + \hat{s}_i + \bar{t}_{ij}, & \\
\quad a_{i\gamma\varsigma-1} + \bar{s}_i + \bar{t}_{ij} + \hat{t}_{ij}, & \\
\quad a_{i\gamma-1\varsigma-1} + \bar{s}_i + \hat{s}_i + \bar{t}_{ij} + \hat{t}_{ij} \Big\} & j > 1,\ i = j-1,\ 0 < \gamma \le \Gamma,\ 0 < \varsigma \le \Upsilon. \quad \text{(A.3g)}
\end{cases}
$$

Case (A.3a) indicates that each vehicle starts its trip form the depot v_0 at time 0. As no service is required at the depot, each vehicle arrives at its first customer v_1 (i.e., $j = 1$) at time $\bar{t}_{0j}$ if arc $(0, j)$ takes its nominal service time, leading to case (A.3b); otherwise, leading to case (A.3c). For the next few nodes $v_j, j > 1$ along the route, there exist four cases for calculating the arrival time:

- Case (A.3d) denotes that no uncertainty occurs in the given route as $\gamma = 0$ and $\varsigma = 0$. In this case, the service times at all customers before v_j take their nominal values, and the travels times on all arcs before v_j also take their nominal values. Then the arrival time at v_j is equal to the arrival time at its predecessor plus the nominal service time there and the nominal travel time between them.

- Case (A.3e) suggests that only service time uncertainties occur along the given route as $\varsigma = 0$ and $\gamma > 0$, which is identical to case (15d) in Section 3.3.2.

- Case (A.3f) indicates that only travel time uncertainties occur along the given route as $\gamma = 0$ and $\varsigma > 0$. We can derive the two terms in (A.3f) using the same method as for case (15d).

- Case (A.3g) denotes that both types of uncertainties occur along the given route as $0 < \gamma \leq \Gamma$ and $0 < \varsigma \leq \Upsilon$. For the γ served customers and ς traversed arcs before v_j that have attained their worst-case values, we need to consider four situations with respect to the composition of these customers and arcs for calculating $a_{j\gamma\varsigma}$. Specifically, there are (i) γ customers and ς arcs; (ii) $\gamma - 1$ customers and ς arcs; (iii) γ customers and $\varsigma - 1$ arcs; and (iv) $\gamma - 1$ customers and $\varsigma - 1$ arcs, which attain their worst-case values before v_{j-1}. The four terms in (A.3g) respectively correspond to the four situations. Finally, $a_{j\gamma\varsigma}$ takes the maximum value among the four terms.

Since $a_{0\gamma\varsigma}$ and $a_{1\gamma\varsigma}$ are known for all $\gamma \leq \Gamma$ and $\varsigma \leq \Upsilon$, we can use equation (A.3) to find $a_{j\gamma\varsigma}$ for all $j > 1$, $0 \leq \gamma \leq \Gamma$ and $0 \leq \varsigma \leq \Upsilon$.

A.3 Modification to the RO Model

The robust formulation of the RTOP-DP that considers both travel time uncertainties and service time uncertainties can be written as follows:

$$\max \sum_{i \in \mathcal{N}_c} p_i y_i - d_i a_{i\Gamma\Upsilon} \tag{A.4}$$

s.t. Constraints (2)–(5), (10), and (11),

$$a_{i\gamma\varsigma} + \bar{s}_i + \bar{t}_{ij} \leq a_{j\gamma\varsigma} + M_{ij}(1 - x_{ij}) \qquad \forall i \in \mathcal{N}_c, j \in \mathcal{N}, 0 \leq \gamma \leq \Gamma, 0 \leq \varsigma \leq \Upsilon, \tag{A.5}$$

$$a_{i\gamma-1\varsigma-1} + \bar{s}_i + \hat{s}_i + \bar{t}_{ij} + \hat{t}_{ij} \leq a_{j\gamma\varsigma} + M_{ij}(1 - x_{ij}) \ \forall i \in \mathcal{N}_c, j \in \mathcal{N}, 1 \leq \gamma \leq \Gamma, 1 \leq \varsigma \leq \Upsilon, \tag{A.6}$$

$$a_{i\gamma\varsigma-1} + \bar{s}_i + \bar{t}_{ij} + \hat{t}_{ij} \leq a_{j\gamma\varsigma} + M_{ij}(1 - x_{ij}) \quad \forall i \in \mathcal{N}_c, j \in \mathcal{N}, 0 \leq \gamma \leq \Gamma, 1 \leq \varsigma \leq \Upsilon, \tag{A.7}$$

$$a_{i\gamma-1\varsigma} + \bar{s}_i + \hat{s}_i + \bar{t}_{ij} \leq a_{j\gamma\varsigma} + M_{ij}(1 - x_{ij}) \quad \forall i \in \mathcal{N}_c, j \in \mathcal{N}, 1 \leq \gamma \leq \Gamma, 0 \leq \varsigma \leq \Upsilon, \tag{A.8}$$

$$a_{i\gamma 0} \geq \bar{t}_{0i} y_i, \qquad \forall i \in \mathcal{N}_c, 0 \leq \gamma \leq \Gamma, \tag{A.9}$$

$$a_{i\gamma\varsigma} \geq (\bar{t}_{0i} + \hat{t}_{0i}) y_i \qquad \forall i \in \mathcal{N}_c, 0 \leq \gamma \leq \Gamma, 1 \leq \varsigma \leq \Upsilon, \tag{A.10}$$

$$a_{i\gamma\varsigma} \leq D_i y_i \qquad \forall i \in \mathcal{N}_c, 0 \leq \gamma \leq \Gamma, 0 \leq \varsigma \leq \Upsilon, \tag{A.11}$$

$$a_{0\gamma\varsigma} \leq T_{\max} \qquad \forall 0 \leq \gamma \leq \Gamma, 0 \leq \varsigma \leq \Upsilon, \tag{A.12}$$

$$a_{i\gamma\varsigma} \geq 0 \qquad \forall i \in \mathcal{N}, 0 \leq \gamma \leq \Gamma, 0 \leq \varsigma \leq \Upsilon. \tag{A.13}$$

The objective function (A.4) maximizes the overall worst-case profit collected by all the vehicles under uncertainties. Constraints (A.5)–(A.8) describe the relationship of arrival times between two adjacent nodes, corresponding to cases (A.3d)–(A.3g), respectively. M_{ij} is a large constant, which can be set as $M_{ij} = D_i + \bar{s}_i + \hat{s}_i + \bar{t}_{ij} + \hat{t}_{ij}$. Constraints (A.9) and (A.10) respectively indicate the lower bound of the arrival time at customer i under the deterministic and the uncertain situations, corresponding to cases (A.3a) and (A.3b). Constraints (A.11) enforce that any selected customer is served before its deadline. Constraints (A.12) ensure that all the vehicles return to the depot no later than $T_{\max}$ under any uncertainty tolerance. Constraints (A.13) define the domains of variables $a_{i\gamma\varsigma}$.

Table B1: Average Performance of the Three Relaxation Techniques In Terms of Solution Quality

			# O (Unbounded)			ub_{root}		
K	$T_{\max}$	$\frac{\bar{N_c}}{K}$	BP-2-En	BP-2	BP-3	BP-2-En	BP-2	BP-3
2	$T_{\max}^{Q1}$	2.3	10	10	10	133.5	133.5	133.5
	$T_{\max}^{Q2}$	4.6	10	10	3	211.7	211.8	211.3
	$T_{\max}^{Q3}$	6.3	10	10	4 (5)	225.3	225.3	225.2
3	$T_{\max}^{Q1}$	2.3	10	10	10	190.3	190.3	190.3
	$T_{\max}^{Q2}$	5.0	10	10	7	273.5	273.5	273.3
	$T_{\max}^{Q3}$	5.3	10	10	5 (3)	273.5	273.5	273.4
4	$T_{\max}^{Q1}$	2.5	10	10	10	243.2	243.2	243.2
	$T_{\max}^{Q2}$	5.1	10	10	8 (1)	308.1	308.1	308.1
	$T_{\max}^{Q3}$	5.1	10	10	5 (5)	300.0	300.0	300.0
5	$T_{\max}^{Q1}$	2.6	10	10	10	289.0	289.1	289.0
	$T_{\max}^{Q2}$	4.7	10	10	8	337.6	337.6	337.6
	$T_{\max}^{Q3}$	4.8	10	10	4 (1)	335.7	335.7	335.7

Appendix B Comparison of Different Relaxations of the ESPPRC

In this section, we evaluate the performance of the *2*-cycle-SPPRC with the enhanced dominance rule (denoted as BP-2-En) by comparing it with the other two relaxations of the ESPPRC, i.e., the *2*-cycle-SPPRC with the basic dominance rule (denoted as BP-2) and the *3*-cycle-SPPRC with the basic dominance rule (denoted as BP-3). We note that the enhanced dominance rule presented in Section 4.3 does not apply when $k > 2$. The computing time limit of the B&P is set to 3600 seconds. We conduct the experiments using 25-customer instances and report the average results in Tables B1 and B2, where results are grouped by the number of vehicles K and the time horizon $T_{\max}$. There are ten instances with different values of Γ and ϵ in each group. In these two tables, $\frac{\bar{N_c}}{K}$ refers to the average number of customers per route in the optimal solution produced by the BP-2-En.

Table B1 reports the performance of the three relaxation techniques in terms of solution quality, where #O denotes the number of instances solved to optimality; the value in the brackets next to #O is the number of instances that the BP-3 cannot provide an upper bound within the time limit; and ub_{root} is the upper bound provided by each method after solving the root node of the B&B tree. Table B1 shows that both the BP-2-En and the BP-2 can solve all the instances to optimality for any value of $T_{\max}$. However, the BP-3 only finds optimal solutions for all the instances with a short time horizon $T_{\max}^{Q1}$. For these instances, the average number of customers per route $\frac{\bar{N_c}}{K}$ is smaller than 3, indicating that the 3-cycle paths are seldom generated during the labeling process of the BP-3. In this situation, the BP-3 is almost equivalent to the BP-2; thus, the time differences between these two methods are minor. When $\frac{\bar{N_c}}{K}$ gets larger with $T_{\max}$, the BP-3 solves fewer instances to optimality. Moreover, it cannot even produce an upper bound for some instances when $\frac{\bar{N_c}}{K} \geq 5$. The purpose of considering a larger value of k in the k-cycle-SPPRC is to tighten the upper bounds by avoiding generating more routes with cycles; however, as shown in Table B1, the upper bounds obtained from the BP-3 at the root nodes are almost the same as those generated by the other two methods.

Table B2: Average Computing Times and Explored B&B Tree Nodes of the Three Relaxation Techniques

			$Time$ (s)			$Time_L$ (s)			#Node		
K	$T_{\max}$	$\frac{\bar{N}_c}{K}$	BP-2-En	BP-2	BP-3	BP-2-En	BP-2	BP-3	BP-2-En	BP-2	BP-3
2	$T_{\max}^{Q1}$	2.3	0.6	0.5	2.5	0.2	0.2	1.2	1	1	1
	$T_{\max}^{Q2}$	4.6	234.2	1057.6	2900.1	0.6	2.9	117.6	118	133	21
	$T_{\max}^{Q3}$	6.3	26.2	169.4	3239.6	0.9	7.6	248.2	6	6	3
3	$T_{\max}^{Q1}$	2.3	0.8	0.5	2.5	0.2	0.2	1.2	1	1	1
	$T_{\max}^{Q2}$	5.0	80.7	475.2	2188.0	1.2	6.3	176.6	23	25	5
	$T_{\max}^{Q3}$	5.3	79.7	527.3	2799.7	1.0	6.2	226.0	18	18	2
4	$T_{\max}^{Q1}$	2.5	3.1	1.6	6.4	0.2	0.2	1.1	4	4	4
	$T_{\max}^{Q2}$	5.1	31.4	231.6	1761.8	0.7	7.3	135.5	5	4	2
	$T_{\max}^{Q3}$	5.1	28.6	148.1	2745.6	1.0	6.4	193.8	2	2	1
5	$T_{\max}^{Q1}$	2.6	34.0	21.5	27.6	0.2	0.2	0.8	67	89	37
	$T_{\max}^{Q2}$	4.7	28.8	148.5	2292.5	0.7	3.5	88.6	6	8	6
	$T_{\max}^{Q3}$	4.8	37.6	124.1	2972.4	0.7	3.3	172.7	7	7	4

Table B2 reports the average results regarding the computing time $Time$ of solving an instance, the time consumption of each call of the labeling algorithm $Time_L$, and the number of nodes #Node explored in the B&B tree. Table B2 shows that when $T_{\max}$ equals to $T_{\max}^{Q1}$, the BP-3 is as competitive as the other two methods, solving all the instances to optimality in a short time. This is because the average number of customers per route is less than 3. When the time horizon becomes longer, the BP-2-En consumes the least computing time, significantly less than the other two methods. The increased computing times of the BP-2 and the BP-3 are mainly due to the longer time required by the labeling algorithm, as indicated in columns $Time_L$. Once more times are spent at a B&B tree node, fewer nodes could be explored within a given time limit, as shown in columns #Node. Thus, we conclude that the BP-2-En is the most efficient method for solving the pricing problem among the three relaxations.