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 vehicle routes 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 tractable robust formulation for the problem, which utilizes dynamic programming recursive equations to calculate the worst-case arrival times at customers along a given route. To solve the robust model, we develop a branch-and-price (B&P) algorithm for exact solutions and a tabu search (TS) algorithm for approximation solutions. Numerical tests show that our B&P algorithm can solve most instances with 100 customers to optimality within 25 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

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 while respecting the route duration constraint. A natural extension of the OP is the team orienteering problem (TOP), which generalizes the OP to a multi-route case. 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 (Chao
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 last-mile distribution, search, and rescue 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. 2021).

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 congestions, uncertain wait 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 take uncertainties into account proactively, in order 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, if the actual service times at some selected customers are longer than the estimated values, the service at other customers might be postponed and the duration of a route might be prolonged, leading to the violations of deadline 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 the uncertain service times, which heavily depend on the conditions where customers are. Different conditions (e.g., the extent of damage of equipment in the maintenance scheduling problem and the extent of injury of a customer 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 whereas 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. Since it is often difficult to identify a probability distribution for the uncertain service time due to limited historical data in most applications, especially in the context of disaster relief operations, we adopt the 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 TOP-DP.

1.1. Our Contributions
This paper studies a robust team orienteering problem with decreasing profits (RTOP-DP), where the uncertain service times at customers are characterized by a budget uncertainty set. The RTOP-DP aims to maximize the collected profit and also guarantee the feasibility of scheduled routes for any uncertainty realization within the uncertainty set. In the RTOP-DP, the deadline constraint indicates that there is an upper bound on the arrival time at each customer. Meanwhile, a lower bound is also implicitly imposed, which is the travel time from the depot to the customer. These bounds lead to a time window constraint for each customer. Thus, the RTOP-DP is indeed a variant of the TOP with time window (TOPTW), where the customer profits are time-decreasing and the service times are uncertain. When implementing the RO method, the time window constraints must be explicitly expressed in the formulation, resulting in a larger number of constraints in the robust counterpart model and a significant increase of computational complexity, compared to the TOPTW, which is already NP-hard. To explicitly incorporate uncertainties into the mathematical model whereas not significantly increase the computational complexity, we
employ the dynamic programming recursive equations introduced by Munari et al. (2019) to calculate the worst-case arrival time at each node. We then propose a tractable model for the RTOP-DP and develop both exact and heuristic algorithms to solve it. Our work makes the following contributions to the literature:

- We formally introduce the RTOP-DP, which is a new generalization of the TOPTW in the context of uncertainty. To the best of our knowledge, this work is the first to address a stochastic TOPTW using the RO technique.

- We present a tractable model for the RTOP-DP by using dynamic programming recursive equations to compute the worst-case arrival times under all anticipated realizations of uncertain service times. To solve the model exactly, we develop a branch-and-price (B&P) algorithm based on a route-based reformulation of the RTOP-DP, where a tailored labeling algorithm is developed to solve the robust shortest path problem with resource constraints. To solve large-size instances efficiently, we further develop a tabu search (TS) algorithm.

- 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 25 minutes and that the TS algorithm can find high-quality solutions within a few seconds. Moreover, 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 wait 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 target 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 Monto 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 the RO method to solve the OPSW, where the weights refer to the travel times. Evers et al. (2014b) is the first to use a 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 proposed to select reward nodes and determine 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 Yanıkoglu (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 a 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 develops a hybrid heuristic algorithm.

Since the RTOP-DP is a variant of the TOPTW, its most relevant problem is the uncertain variant of the OP with time windows (OPTW), which has gained in popularity in recent years. Evers et al. (2014a) is 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 wait times and address the OPTW with uncertain wait 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 solving 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 Yaniçoğlu (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 by using sampling or simulation in heuristic algorithms. However, in an exact method framework, it is extremely time-consuming to deal with the arrival time-related constraints, i.e., the Miller-Tucker-Zemlin (MTZ) constraints (Miller et al. 1960). To overcome these challenges, we use the RO method and the dynamic programming recursive equations to tackle the RTOP-DP. Besides a heuristic method, we also develop an exact algorithm to solve the robust model.

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 $G = (\mathcal{N}, \mathcal{A})$, 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. 2021), 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 $[0, T_{\text{max}}]$. Then the deadline at the depot or the time horizon of the considered problem is $T_{\text{max}}$. To maximize the overall profit, we need to make two sets of decisions: the assignment of customers to vehicles and the service sequence of customers along each route. To this end, we define the following decision variables:

- $x_{ijk} = 1$ if vehicle $k \in \mathcal{K}$ traverses directly from node $i \in \mathcal{N}$ to node $j \in \mathcal{N}$, 0 otherwise;
- $y_{ik} = 1$ if customer $i \in \mathcal{N}_c$ is served by vehicle $k \in \mathcal{K}$, 0 otherwise;
- $a_i$ is 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:

$$\begin{aligned}
\max & \sum_{k \in \mathcal{K}} \sum_{i \in \mathcal{N}_c} p_i y_{ik} - d_i a_i \\
\text{s.t.} & \sum_{j \in \mathcal{N}_c} x_{0jk} = 1 & \forall k \in \mathcal{K}, \\
& \sum_{i \in \mathcal{N}_c} x_{i0k} = 1 & \forall k \in \mathcal{K}, \\
& \sum_{j \in \mathcal{N}_c, j \neq i} x_{ijk} = \sum_{j \in \mathcal{N}_c, j \neq i} x_{jik} & \forall i \in \mathcal{N}_c, k \in \mathcal{K}, \\
& \sum_{j \in \mathcal{N}_c, j \neq i} x_{jik} = y_{ik} & \forall i \in \mathcal{N}_c, k \in \mathcal{K}, \\
& \sum_{k \in \mathcal{K}} y_{ik} \leq 1 & \forall i \in \mathcal{N}_c, \\
& a_i + s_i + t_{ij} \leq a_j + M_{ij}(1 - x_{ijk}) & \forall i \in \mathcal{N}_c, j \in \mathcal{N}, k \in \mathcal{K},
\end{aligned}$$
Yu et al.: Robust Team Orienteering Problem with Decreasing Profits
Article submitted to ; manuscript no. (Please, provide the manuscript number!)

\[ a_i \geq t_{0i} \sum_{k \in \mathcal{K}} y_{ik} \quad \forall i \in \mathcal{N}_c, \] (8)

\[ a_i \leq D_i \sum_{k \in \mathcal{K}} y_{ik} \quad \forall i \in \mathcal{N}_c, \] (9)

\[ a_0 \leq T_{\text{max}}, \] (10)

\[ x_{ijk} \in \{0, 1\} \quad \forall i, j \in \mathcal{N}, k \in \mathcal{K}, \] (11)

\[ y_{ik} \in \{0, 1\} \quad \forall i \in \mathcal{N}_c, k \in \mathcal{K}, \] (12)

\[ a_i \geq 0 \quad \forall i \in \mathcal{N}. \] (13)

The objective function (1) maximizes the overall profit collected by all the vehicles. Constraints (2)–(3) ensure that each vehicle starts from and returns to the depot exactly once. Constraints (4) guarantee the flow conservation at each customer. Constraints (5) ensure that variable \( y_{ik} \) is uniquely determined by variables \( x_{ijk} \). Constraints (6) suggest that each customer is served at most once. Constraints (7) are the MTZ constraints, which describe the relationship of arrival times between two adjacent nodes. We set the large constants \( M_{ij} = D_i + s_i + t_{ij} \). Constraints (8) and (9) indicate the lower and upper bounds of arrival times at customers, respectively. Specifically, if customer \( i \) is served by a vehicle, then the earliest arrival time at it must be larger than or equal to the time traveling directly from the depot to it, and the latest arrival time must be earlier than its deadline; otherwise, the arrival time at customer \( i \) is 0. Constraints (10) ensure that all the vehicles return to the depot no later than \( T_{\text{max}} \). Constraints (11)–(13) 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

\[ \tilde{s}_i = \bar{s}_i + \xi_i \hat{s}_i, \xi_i \in [-1, 1], \forall i \in \mathcal{N}_c, \] (14)

where \( \bar{s}_i \) is the nominal service time at customer \( i \). \( \hat{s}_i \) is the largest derivation of service time from its nominal value. \( \xi_i \) is the uncertain parameter residing in \([-1, 1]\). 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

\[ \mathcal{U} = \left\{ \tilde{s} \in \mathbb{R}^{|\mathcal{N}_c|} : \tilde{s}_i = \bar{s}_i + \xi_i \hat{s}_i, \sum_{i \in \mathcal{N}_c} \xi_i \leq \Gamma, 0 \leq \xi_i \leq 1, \forall i \in \mathcal{N}_c \right\}, \] (15)
where parameter $\Gamma$ is called the uncertainty budget, which takes an integer value in the interval $[0, |\mathcal{N}|]$. $\Gamma$ 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 $\xi_i$ is defined to be in the interval $[-1, 1]$ in equation (14), considering that the worst-case scenario is realized only when service times have positive deviations, it is reasonable to assume that $\xi_i$ resides in $[0, 1]$ in the uncertainty set $\mathcal{U}$.

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 $T_{\text{max}}$.

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 $\gamma \leq \Gamma$ customers’ service times have reached their worst-case values. Note that we call $\gamma$ 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:

\[
\begin{align*}
    a_{v_j,\gamma} &= 0, & j = 0, \ 0 \leq \gamma \leq \Gamma, \quad (16a) \\
    t_{v_0v_j}, & j = 1, \ 0 \leq \gamma \leq \Gamma, \quad (16b) \\
    a_{v_{j-1},\gamma} + s_{v_{j-1}} + t_{v_{j-1}v_j}, & j > 1, \ \gamma = 0, \quad (16c) \\
    \max \left\{ a_{v_{j-1},\gamma} + s_{v_{j-1}} + t_{v_{j-1}v_j}, a_{v_{j-1},\gamma-1} + s_{v_{j-1}} + \hat{s}_{v_{j-1}} + t_{v_{j-1}v_j} \right\}, & j > 1, \ 0 < \gamma \leq \Gamma. \quad (16d)
\end{align*}
\]

Case (16a) 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 $t_{v_0v_1}$ regardless of the value of $\gamma$, leading to case (16b). 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., $s_{v_{j-1}}$) and the travel time $t_{v_{j-1}v_j}$.
between them, leading to case (16c). When $0 < \gamma \leq \Gamma$, i.e., there are $\gamma$ 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 $\gamma$ customers for calculating the value of $a_{v_j\gamma}$. The first situation is that there are $\gamma$ 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 (16c) for the former situation, and $a_{v_j\gamma} = a_{v_j-1\gamma-1} + \bar{s}_{v_j-1} + \hat{s}_{v_j-1} + t_{v_j-1v_j}$ 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 (16d). Since $a_{v_0\gamma}$ and $a_{v_1\gamma}$ are known for all $0 \leq \gamma \leq \Gamma$, we can use the recursive equation (16) to find $a_{v_j\gamma}$ for all $j > 1$ and $0 \leq \gamma \leq \Gamma$.

As a vehicle can start its service immediately once arriving at a customer, we can obtain $a_{v_j\gamma}$ alternatively by adding the $\gamma$-largest service time derivations of the customers served before $v_j$ to the nominal arrival time at $v_j$ (Lee et al. 2012), i.e.,

$$a_{v_j\gamma} = \sum_{i=0}^{j-1} (t_{vi_{i+1}} + \bar{s}_i) + \max_{\varphi \subseteq \{1, \ldots, j-1\}, |\varphi| \leq \gamma} \sum_{i \in \varphi} \hat{s}_{vi} \quad \text{(17)}$$

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 $D_{v_{m+1}} = T_{\text{max}}$ for the depot $v_{m+1}$.

### 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. Then we can formulate the RTOP-DP as

$$\max \sum_{k \in \mathcal{K}} \sum_{i \in \mathcal{N}_c} p_i y_{ik} - d_i a_{i\Gamma} \quad \text{(18)}$$

s.t. Constraints (2)–(6) and (11)–(12),

$$a_{i\gamma} + \bar{s}_i + t_{ij} \leq a_{j\gamma} + M'_{ij}(1 - x_{ijk}) \quad \forall i \in \mathcal{N}_c, j \in \mathcal{N}, k \in \mathcal{K}, \gamma \in \{0, \ldots, \Gamma\}, \quad \text{(19)}$$

$$a_{i\gamma-1} + \bar{s}_i + \hat{s}_i + t_{ij} \leq a_{j\gamma} + M''_{ij}(1 - x_{ijk}) \quad \forall i \in \mathcal{N}_c, j \in \mathcal{N}, k \in \mathcal{K}, \gamma \in \{1, \ldots, \Gamma\}, \quad \text{(20)}$$

$$a_{i\gamma} \geq t_{0i} \sum_{k \in \mathcal{K}} y_{ik} \quad \forall i \in \mathcal{N}_c, \gamma \in \{0, \ldots, \Gamma\}, \quad \text{(21)}$$

$$a_{i\gamma} \leq D_i \sum_{k \in \mathcal{K}} y_{ik} \quad \forall i \in \mathcal{N}_c, \gamma \in \{0, \ldots, \Gamma\}, \quad \text{(22)}$$

$$a_{0\gamma} \leq T_{\text{max}} \quad \forall \gamma \in \{0, \ldots, \Gamma\}, \quad \text{(23)}$$
\[ a_{i\gamma} \geq 0 \quad \forall i \in \mathcal{N}, \gamma \in \{0, \ldots, \Gamma\}. \tag{24} \]

The objective function (18) maximizes the overall worst-case profit collected by all the vehicles. Constraints (19) and (20) describe the relationship of arrival times between two adjacent nodes without and with one unit increment of uncertainty tolerance, respectively. Specifically, constraints (19) represent the case where the uncertainty tolerance keeps unchanged from nodes \(i\) to \(j\). We set the large constants \(M'_{ij} = D_i + \bar{s}_i + t_{ij}\). Whereas constraints (20) denote the case where the uncertainty tolerance increases by one unit from nodes \(i\) to \(j\), 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 (7) 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 window, 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 (21) and (22) indicate the lower and upper bounds of the arrival time at customer \(i\). Constraints (23) ensure that all the vehicles return to the depot no later than \(T_{\text{max}}\) under any uncertainty tolerance. Constraints (24) define the domains of variables \(a_{i\gamma}\).

Although the RTOP-DP model can be directly solved by commercial solvers like CPLEX and Gurobi, significant computational resources will be required as the number of customers increases. In the literature of the OP and its variants, it is a common practice to apply some B&C based methods for effectively solving the mathematical models (Fischetti et al. 1998, Tang and Miller-Hooks 2005, Erdoğän and Laporte 2013, Angelelli et al. 2017, Yu et al. 2021). Our preliminary tests show that a B&C based method cannot find the optimal solutions for most small-size RTOP-DP instances within one hour, thus more efficient exact algorithm is needed.

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 (C&G) framework in Sections 4.2 and 4.3, respectively. Finally, Section 4.4 presents the branching scheme used in the B&P algorithm.
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 $\mu_r$ 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 $\pi_{ir}$ indicates the number of times route $r$ serves customer $i$. We are interested in finding a robust feasible solution $\text{sol} = \{r_1, r_2, \ldots, r_K\}$, 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:

$$
[P]: \quad \text{max} \sum_{r \in \mathcal{R}} p_r \theta_r \quad \tag{25}
$$

s.t. \quad \sum_{r \in \mathcal{R}} \theta_r \leq K, \quad \tag{26}

\quad \sum_{r \in \mathcal{R}} \pi_{ir} \theta_r \leq 1 \quad \forall i \in \mathcal{N}_c, \quad \tag{27}

\quad \theta_r \in \{0, 1\} \quad \forall r \in \mathcal{R}. \quad \tag{28}

The objective function (25) maximizes the profit by selecting routes. Constraint (26) guarantees that at most $K$ routes are selected in the final solution. Constraints (27) ensure that each customer is served at most once. Constraints (28) 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 C&G 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 C&G 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 C&G. Starting from solving the [LP] with a reduced subset $\hat{\mathcal{R}} \subseteq \mathcal{R}$, which is usually called as the restricted master problem [RMP] in the C&G, the primal solution $\theta = (\theta_1, \ldots, \theta_{|\hat{\mathcal{R}}|})$ and the dual solution $\lambda = (\lambda_0, \lambda_1, \ldots, \lambda_{|\mathcal{N}_c|})$ are obtained, where $\lambda_0$ is a nonnegative dual variable for constraint (26) and $\lambda_i$ is a nonnegative dual variable associated with constraints (27) indexed by customer $i \in \mathcal{N}_c$. 
If the optimal solution $\theta$ 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 $\theta$ to the [LP] is equivalent to checking if the dual solution $\lambda$ is optimal to the dual problem of the [LP]. We write the dual of the [LP] as

$$\text{[DLP]}: \min K\lambda_0 + \sum_{i \in N_c} \lambda_i \quad (29)$$

s.t. $\lambda_0 + \sum_{i \in N_c} \pi_{ir}\lambda_i \geq p_r \quad \forall r \in R, \quad (30)$

$$\lambda_i \geq 0 \quad \forall i \in N. \quad (31)$$

If there exists any robust feasible route $r' \in R$ violating constraints (30), the current dual solution $\lambda$ is not optimal for the [DLP], neither is $\theta$ optimal for the [LP]. Moreover, it implies that moving route $r'$ to the subset $\hat{R}$ 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 $\lambda$, we define the reduced cost of a robust feasible route $r$ as

$$c_r(\lambda) = \lambda_0 + \sum_{i \in N_c} \pi_{ir}\lambda_i - p_r. \quad (32)$$

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

$$c_r(\lambda) = \lambda_0 + \sum_{i \in N_c} \pi_{ir}\lambda_i - \sum_{i \in N_c} (p_i - d_i a_i \Gamma) \pi_{ir} = \lambda_0 + \sum_{i \in N_c} \pi_{ir}(\lambda_i - p_i + d_i a_i \Gamma).$$

Let $\beta_{ijr}$ 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

$$c_r(\lambda) = \sum_{(i,j) \in A} \beta_{ijr}(\lambda_j - p_j + d_j a_j \Gamma). \quad (33)$$

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

The pricing problem corresponds to the well-known elementary shortest path problem with resource constraints (ESPPRC) on graph $G = (N, A)$. The ESPPRC is a well-known
NP-hard problem, which 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, which is also utilized in our study.

In the labeling algorithm, each partial path from the depot to a given node $i \in \mathcal{N}$ is represented by a label $L_i = (pre_i, c_i, a_{i0}, \ldots, a_{i\Gamma})$, where $pre_i$ is the direct proceeder of node $i$. $c_i$ is the reduced cost up to node $i$. $a_{i\gamma}$ 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 $L_0 = (null, 0, 0, \ldots, 0)$. Then, the algorithm extends label $L_i$ to node $j \in \mathcal{N} \setminus \{i, Pre_i\}$, generating a new label $L_j$ according to the following relationships:

\begin{align}
pre_j &= i, \\
c_j &= c_i + \lambda_j - p_j + d_j a_{j\Gamma}, \\
a_{j0} &= a_{i0} + t_{ij} + \bar{s}_i, \\
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\}. 
\end{align}

The resulting path associated with label $L_j$ 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 commonly used dominance rule (adapted from that for the SPPRC) for the $2$-cycle-SPPRC only applies to labels having the 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 their dominance rule to work for our robust 2-cycle-SPPRC.

We first define the dominance relation between two labels in the context of robust SPPRC. Given two labels $L_i^1$ and $L_i^2$ ending at the same node $i$, $L_i^2$ is dominated by $L_i^1$ if the following conditions are satisfied: (i) $c_i^1 \leq c_i^2$, and (ii) $a_i^{1\gamma} \leq a_i^{2\gamma}, \forall \gamma \in \{0,1,\ldots,\Gamma\}$.

Next, we introduce the three types of dominant labels for the robust 2-cycle-SPPRC.

**Definition 1.** A label $L_i = (pre_i, c_i, a_i^0, \ldots, a_i^\Gamma)$ is defined as **strongly dominant** if it is not dominated by any other labels, and $(a_i^0 + \bar{s}_i + t_{pre_i}) > D_{pre_i}$ holds or $\max\{a_i^{\gamma} + \bar{s}_i + t_{pre_i}, a_i^{\gamma-1} + \bar{s}_i + \hat{t}_{pre_i}\} > D_{pre_i}$ holds for each $\gamma \in \{1, \ldots, \Gamma\}$.

**Definition 2.** A label $L_i = (pre_i, c_i, a_i^0, \ldots, a_i^\Gamma)$ is defined as **semistrongly dominant** if it is not dominated by any other labels and it is not strongly dominant.

**Definition 3.** A label $L_i = (pre_i, c_i, a_i^0, \ldots, a_i^\Gamma)$ is defined as **weakly dominant** if it is only dominated by semistrongly dominant labels that share the identical predecessor node but not $pre_i$.

Due to the deadline constraint, a strongly dominant label should not be extended to its predecessor node, in such case, a 2-cycle operation cannot be performed. Thus, any label that is dominated by a strongly dominant label can be discard. In addition, any label that does not belong to the aforementioned three types can be discarded. We note that a weakly dominant label should be kept, because a semistrongly dominant label cannot be extended to its predecessor due to the limit of 2-cycle, instead its weakly dominant label can be extended to this predecessor.

### 4.4. Branching Scheme

The branching scheme for the TOP is different from that for the vehicle routing problem (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 $\hat{\theta}$ 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, $i' = \arg\min_{i \in \mathcal{N}_{c}(\hat{\theta})} \{|\sum_{r \in \mathcal{R}} \pi_{ir} \hat{\theta}_{r} - 0.5|\}$, where $\mathcal{N}_{c}(\hat{\theta})$ consists the customers who are served a positive number of times in the solution $\hat{\theta}$. Then we derive two branches on this customer. In the first one, customer $i'$ must be served, that is, $\sum_{r \in \mathcal{R}} \pi_{i'r} \theta_{r} = 1$. In the second one, customer $i'$ cannot be served, that is, $\sum_{r \in \mathcal{R}} \pi_{i'r} \theta_{r} = 0$.

The second rule is based on the fractional arcs when all the 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 $\hat{\theta}$, i.e., $(i', j') = \arg\min_{(i,j) \in \mathcal{A}(\hat{\theta})} \{|\sum_{r \in \mathcal{R}} \beta_{ijr} \hat{\theta}_{r} - 0.5|\}$, where $\mathcal{A}(\hat{\theta})$ is the set of arcs that are traversed by vehicles, i.e., $\mathcal{A}(\hat{\theta}) = \{(i,j) \in \mathcal{A} | \sum_{r \in \mathcal{R}} \beta_{ijr} \hat{\theta}_{r} > 0\}$. Instead of a binary branching which is commonly used for fractional arcs in the VRP, we consider two cases based on constraints (27) associated with customers $i'$ and $j'$ to conduct different branching rules. 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 the arc $(i', j')$; the second branch forces to serve customer $i'$, but forbids the use of this arc; and the third branch forbids to serve customer $i'$.

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 $\Gamma_{\text{max}}$ consecutive iterations. We set $\Gamma_{\text{max}} = 50$ based on some preliminary tests.
Algorithm 1 Framework of the tabu search algorithm

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

To avoid searching the same space for several times, we consider a tabu list $\mathcal{L}$ 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 $\eta$ iterations, where $\eta$ represents the tabu tenure. We put the forbidden move and its tabu tenure into the tabu list $\mathcal{L}$. The value of $\eta$ associated with each forbidden move is updated according to the quality of the found solution. In particular, the initial value of $\eta$ 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 $\iota_k$ and assume that an unserved customer $i$ is inserted after $\iota_k$. Then $\Delta t_{\iota_k,i} = t_{\iota_k,i} + t_{i0} - t_{\iota_k0}$ 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. 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 $r_k \in \text{sol}$. If it is infeasible to insert customer $i$ into route $r_k$, i.e., vehicle $k$ cannot
Algorithm 2 Initialization of a robust feasible solution

1: **Input:** The customer set $\mathcal{N}_c = \{1, \ldots, n\}$, the vehicle set $\mathcal{K} = \{1, \ldots, K\}$ and the time horizon $T_{\text{max}}$.

2: Let $\text{sol} = \{r_k = \{0,0\}| \forall k \in \mathcal{K}\}$; denote the last customer of every route $r_k$ as $i_k$, and the arrival time at the ending depot as $\tau_k$.

3: Initial $i_k \leftarrow 0$, and $\tau_k \leftarrow 0$.

4: repeat

5: for all $i \in \mathcal{N}_c$ do

6: for all $k \in \mathcal{K}$ do

7: Insert $i$ into route $r_k$, and place it between $i_k$ and the ending depot 0, then $a_{ik} = a_{ik} + \bar{s}_{ik} + \hat{s}_{ik} + t_{ik}$, and the time arriving at the ending depot is $\tau_k = a_{ik} + \bar{s}_{i} + \hat{s}_{i} + t_{i0}$.

8: if $a_{ik} < D_i$ and $\tau_k \leq T_{\text{max}}$ then

9: $\omega_{ik} = \frac{p_i}{s_{i,\text{ik}}}$

10: else

11: $\omega_{ik} = 0$

12: end if

13: end for

14: end for

15: Let $\zeta = \max_{i \in \mathcal{N}_c, k \in \mathcal{K}} \{\omega_{ik}\}$.

16: if $\zeta > 0$ then

17: Let $(i^*, k^*) \leftarrow \arg\max_{i \in \mathcal{N}_c, k \in \mathcal{K}} \{\omega_{ik}\}$, update $\tau_{k^*} \leftarrow a_{i^*k^*} + \bar{s}_{i^*} + \hat{s}_{i^*} + t_{i^*0}$, $i_{k^*} \leftarrow i^*$, $r_{k^*} = \{0, \ldots, i^*, 0\}$, $\mathcal{N}_c \leftarrow \mathcal{N}_c \setminus i^*$.

18: end if

19: until $\zeta = 0$

20: **Output:** $\text{sol}$.

return to the ending depot before $T_{\text{max}}$ 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.

• **$\lambda$-Exchange**: Change the locations of two chains of $\lambda$ 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 edges from two routes and replace them with other two edges.

• **Or-opt**: Relocate a chain of $\sigma$, $\sigma \in \{1, 2, 3\}$ consecutive customers in the same route.

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 $\psi$ in the interval $[0, 1]$. If $\psi \geq 0.5$, 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 exact algorithms are 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 the instance sets based on Solomon’s type C101 and R101 data sets (Solomon 1987), where the customers are located in a cluster (C) and in random (R), respectively. In each instance, the coordinates and the demand $p_i$ (the profit in our problem) are directly taken from Solomon’s instances. The travel time $t_{ij}$ on arc $(i, j)$ is set to the Euclidean distance and rounded to the nearest hundredth. The first node is taken as the depot and
the next $n$ nodes are taken as the customers. 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 $\bar{s}_i = 50 + 2p_i$. We consider instances of different sizes, i.e., with 25, 50, and 100 customers. Instances with 25 and 50 customers are generated by extracting the first 25 and 50 customers from the instances with 100 customers, respectively. We let the time horizon $T_{\text{max}}$ take three values for each instance, which are the first, second, and third quartile of all customers’ deadlines in that instance, and these values are rounded up to integers. We denote them as $T_{\text{max}}^{Q1}$, $T_{\text{max}}^{Q2}$, and $T_{\text{max}}^{Q3}$. 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 derivation of customer $i$ as $\hat{s}_i = \epsilon \bar{s}_i$, where $\epsilon$ is a proportion belonging to the set $\{10\%, 25\%, 50\%\}$. The uncertainty budget $\Gamma$ takes values from the set $\{0, 1, 5, 10\}$. Thus, there are a total of 720 instances based on the type of instances and different values of $(\epsilon, \Gamma, n, T_{\text{max}}, K)$, among which 648 instances are associated with the robust model ($\Gamma > 0$) and 72 instances are for the deterministic model ($\Gamma = 0$).

6.2. Performance of Exact Algorithms

In this section, we evaluate the performance of the B&P algorithm by comparing it with a Benders branch-and-cut (BBC) algorithm. As mentioned in Section 3, the RTOP-DP is formulated as a mixed-integer linear programming model, which can be directly solved by CPLEX; however, the MTZ constraints (19) and (20) make it extremely time-consuming to do so. Therefore, we employ a BBC method, where these two groups of constraints are replaced by the general subtour elimination constraints, and the objective function and the constrains related to the arrival times are projected to the Benders subproblem. Yu et al. (2021) propose the BBC for the team orienteering problem with time-varying profit, where the mathematical model has the same structure as ours. We briefly introduce the BBC in Appendix A and interested readers are referred to Yu et al. (2021) for more details.

We test the B&P and the BBC on the instances with 25 customers with a time limit of 3600s. Table 2 presents the average results, 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 (also known as the best found objective value) $z_{lb}$, the upper bound $z_{ub}$, the percentage gap measured as $(z_{ub} - z_{lb})/z_{lb} \times 100$, and the computing time.
in seconds. Statistical results of cuts generated in the BBC process are provided in Table B1 in Appendix B.

<table>
<thead>
<tr>
<th>Table 2</th>
<th>Average Results of the B&amp;P and the BBC Algorithms for Small-size Instances with 25 Customers</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Gamma )</td>
<td>#</td>
</tr>
<tr>
<td>0</td>
<td>24</td>
</tr>
<tr>
<td>1</td>
<td>72</td>
</tr>
<tr>
<td>5</td>
<td>72</td>
</tr>
<tr>
<td>10</td>
<td>72</td>
</tr>
<tr>
<td>Avg.</td>
<td>60</td>
</tr>
</tbody>
</table>

Table 2 shows that the B&P significantly outperforms the BBC. In particular, the B&P can solve all the instances to optimality with an average computing time of 37.00 seconds, whereas the BBC can only solve 19 out of 240 instances to optimality. The average optimality gap generated by the BBC is 76.07% for all the instances, and it even goes up to 99.51% for the case of \( \Gamma = 10 \). Notably, the average gap and the computing time of the BBC increase with \( \Gamma \), whereas the average computing time of the B&P decreases in general. The reason for this phenomenon is that a larger value of \( \Gamma \) increases the probability of deadline violations at customers along a route. In this case, the number of Benders feasibility cuts would grow as shown in Table B1, which accounts for a large proportion of all the generated cuts, leading to a longer computing time. Whereas, in the B&P, fewer path extensions are robust feasible when \( \Gamma \) takes a larger value, reducing the computing time of the pricing problem, which is a significant part of the B&P algorithm.

6.3. Performance of Tabu Search Algorithm

In this section, we provide results of the B&P for medium-size and large-size instances and also evaluate the performance of the TS algorithm. For the TS, we run it 10 times for each instance. We present the average results in Table 3, where column \( z_{lb} \) Time refers to the time that the B&P finds the lower bound \( z_{lb} \). For the TS algorithm, we report the best, average, and worst objective values, the relative standard derivation (RSD) among 10 runs, the percentage gap \( \text{Gap}_t \) between \( z_{ub} \) and \( \text{Avg.} \text{Obj} \) (i.e., \( \text{Gap}_t = (z_{ub} - \text{Avg.} \text{Obj})/\text{Avg.} \text{Obj} \times 100 \)), and the total computing time of 10 runs.

From Table 3, we observe that for instances with 50 customers, the B&P can solve 223 out of 240 instances to optimality within 839.89 seconds (about 14 minutes). For instances with
Table 3 Average Results of the B&P and the TS Algorithms

| |N| |Γ |# |#O|z\textsubscript{lb} |z\textsubscript{ub} |Gap (%) |Time (s) |z\textsubscript{lb} |Time (s) |B&P|TS|
|---|---|---|---|---|---|---|---|---|---|---|---|---|
|25 |0 |24 |24 |231.55 |231.55 |0.00 |47.82 |22.22 |231.29 |230.37 |229.24 |0.37 |0.12 |8.6 |
| |1 |72 |72 |221.18 |221.18 |0.00 |37.90 |20.89 |221.15 |220.84 |220.00 |0.18 |0.02 |9.18 |
| |5 |72 |72 |208.27 |208.27 |0.00 |20.09 |9.93 |208.16 |208.04 |207.69 |0.07 |0.04 |7.07 |
| |10 |72 |72 |208.23 |208.23 |0.00 |20.54 |10.58 |208.23 |207.88 |207.41 |0.14 |0.00 |7.08 |
|50 |0 |24 |20 |297.32 |297.38 |0.02 |839.89 |190.00 |296.60 |294.72 |293.02 |0.36 |0.20 |14.15 |
| |1 |72 |67 |279.93 |279.97 |0.01 |563.93 |142.15 |279.63 |278.45 |276.83 |0.35 |0.09 |15.73 |
| |5 |72 |68 |255.18 |255.22 |0.01 |425.72 |119.84 |254.87 |254.21 |253.15 |0.25 |0.12 |11.89 |
| |10 |72 |68 |255.03 |255.06 |0.01 |358.12 |124.76 |254.73 |254.21 |253.25 |0.22 |0.10 |12.17 |
|100 |0 |24 |17 |366.27 |368.35 |0.13 |1513.25 |912.06 |364.55 |362.85 |361.16 |0.34 |0.74 |15.92 |
| |1 |72 |68 |340.17 |341.57 |0.25 |1224.18 |578.87 |340.33 |338.46 |336.05 |0.47 |0.28 |23.71 |
| |5 |72 |71 |305.65 |306.32 |0.13 |763.49 |304.00 |305.54 |304.34 |302.57 |0.35 |0.28 |17.72 |
| |10 |72 |70 |305.61 |305.91 |0.06 |1163.62 |316.03 |305.08 |303.85 |302.27 |0.35 |0.29 |17.57 |
|Avg. |60 |57.4 |267.76 |268.08 |0.06 |537.79 |200.18 |267.52 |266.62 |265.37 |0.27 |0.16 |13.50 |

100 customers, 226 out of 240 instances are optimally solved by the B&P within 1513.25 seconds (about 25 minutes). Moreover, the optimality gaps are quite small—within 0.02% for instances with 50 customers and 0.25% for instances with 100 customers. In addition, the results in column \( z_{lb} \) Time suggest that the B&P can find the 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 Avg. Obj of the TS is 0.16%. And 10 runs of TS only consume 13.50 seconds, which is about 2.5% of the computing time of the B&P (537.79 seconds). We further notice that the TS is also robust as the RSD is only 0.27% 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. Thus, we can use the TS algorithm to solve large-size instances of the RTOP-DP.

6.4. 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 \( \bar{s}_i, i \in N_c \)
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, i.e., 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 $\frac{visN_c}{K}$ under different values of $\Gamma$ and $\epsilon$ in Table 4, where each row provides the average results 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 4, 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 8.75\% for all the cases while the $Risk$ is reduced to 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 86.09\%. 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 70\% for most cases. (2) The $Risk$ of robust solutions is closely related to the uncertainty budget $\Gamma$ and the number of customers in the route $\frac{visN_c}{K}$. In particular, the $Risk$ reduces to 0 when $\Gamma$ is larger than or equal to $\frac{visN_c}{K}$; otherwise, the larger the difference between $\Gamma$ and $\frac{visN_c}{K}$ is, the higher the $Risk$ is. (3) The $PoR$ shows a concavely increasing trend with $\Gamma$, 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 $\Gamma$. Therefore, the uncertainty budget $\Gamma$ 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 randomly chosen instance. 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 $\Gamma$ increases, the $PoR$ increases and the
Table 4  Risk and PoR

| $|N_c|$ | $T_{max}$ | $\Gamma$ | $\epsilon = 10\%$ | $\epsilon = 25\%$ | $\epsilon = 50\%$ |
|-------|---------|---------|--------------|--------------|--------------|
|       |         |         | $PoR$ (%)    | $Risk$ (%)   | $\text{vis}_N$ (%) | $PoR$ (%)    | $Risk$ (%)   | $\text{vis}_N$ (%) | $PoR$ (%)    | $Risk$ (%)   | $\text{vis}_N$ (%) |
| 25    | $T_{Q1}^\text{max}$ | 0       | 43.45       | 5.1          | –             | 58.35       | 5.1          | –             | 73.73       | 5.1          | –             |
|       | $T_{Q2}^\text{max}$ | 1       | 2.58        | 0.0          | 0             | 11.31       | 7.7          | 0             | 68.34       | 4.8          | –             |
|       | $T_{Q3}^\text{max}$ | 5       | 8.75        | 0.0          | 4.5           | 16.42       | 0.0          | 3.4           | 24.31       | 0.0          | –             |
| 50    | $T_{Q1}^\text{max}$ | 0       | 43.53       | 2.4          | –             | 72.85       | 2.4          | –             | 91.40       | 2.4          | –             |
|       | $T_{Q2}^\text{max}$ | 1       | 5.08        | 4.1          | 5.3           | 12.23       | 5.4          | –             | 28.25       | 5.6          | –             |
|       | $T_{Q3}^\text{max}$ | 5       | 6.82        | 0.0          | 5.4           | 16.42       | 0.0          | 3.4           | 24.31       | 0.0          | –             |
| 100   | $T_{Q1}^\text{max}$ | 0       | 32.93       | 6.4          | –             | 53.46       | 6.4          | –             | 70.62       | 6.4          | –             |
|       | $T_{Q2}^\text{max}$ | 1       | 1.39        | 24.59       | 6.3           | 54.72       | 6.3          | 10.50         | 78.86       | 3.6          | –             |
|       | $T_{Q3}^\text{max}$ | 5       | 4.05        | 0.0          | 6.0           | 9.57        | 0.0          | 3.4           | 28.18       | 0.0          | –             |

Risk decreases as expected. When $\Gamma \geq 3$, the robust solution can always guarantee the feasibility of simulation instances, i.e., the risk of deadline violation is 0. Moreover, we can see that when $\Gamma$ 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.

We further explore the characteristics of routes in the optimal solutions under different values of $\Gamma$, 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 derivation (the values in the parenthesis), and the worst-case arrival time at every 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 derivations 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.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{Trade-off between the PoR and the Risk for Instance C25 with $K = 2$, $T_{\max}^{32} = 667$, and $\epsilon = 50\%$}
\end{figure}
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 $\Gamma$.

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 the 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 remain feasible for all the uncertainty real-
izations within the uncertainty set. We utilize dynamic programming recursion equations
to efficiently compute the worst-case arrival time at each selected customer and present a
mixed-integer linear programming model for the problem. To solve the model, we develop
a B&P algorithm for exact solutions and a TS algorithm 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 significantly outperforms the BBC
in the literature, and it can solve most large-size instances to optimality in an accept-
able time frame. Moreover, the TS algorithm can find high-quality solutions for all the
instances within a few seconds, demonstrating the effectiveness of our solution method.
Results further demonstrate that the robust model can effectively reduce the risk of dead-
line violation in simulation tests without significantly compromising the profit, compared
to the deterministic model.

References
Mathematics 41:285–293.


Balcik B, Yanoğlu İ (2020) A robust optimization approach for humanitarian needs assessment planning


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.

230.


Online Supplement

Appendix A  A Benders Branch-and-Cut Algorithm for the RTOP-DP

In the Benders master problem, we keep the routing variables \((x, y)\) and define a new variable \(\varrho_k\) to denote the profit of route \(k\). Then the master problem is written as

\[
\begin{align*}
\max & \quad \sum_{k \in \mathcal{K}} \varrho_k \\
\text{s.t.} & \quad \text{Constraints (2)–(6) and (11)–(12),} \\
& \quad \varrho_k \leq \sum_{i \in \mathcal{N}_c} (p_i - d_i t_{0i}) y_{ik} \quad \forall k \in \mathcal{K}, \\
& \quad \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} t_{ij} x_{ijk} + \sum_{i \in \mathcal{N}} \bar{s}_i y_{ik} \leq T_{\max} \quad \forall k \in \mathcal{K}, \\
& \quad \sum_{(i,j) \in \delta(S)} x_{ijk} \geq 2 y_{hk} \quad \forall S \subset \mathcal{N}, |S| \geq 2, h \in \mathcal{N} \setminus S, k \in \mathcal{K}, \\
& \quad \text{Optimality cuts (A.8) and feasibility cuts (A.9),} \\
& \quad \varrho_k \geq 0 \quad \forall k \in \mathcal{K}.
\end{align*}
\]

The objective function maximizes the total profit. Constraints (A.3) set an upper bound for each \(\varrho_k\). Constraints (A.4) guarantee that the time duration of each route in the nominal case does not exceed the deadline \(T_{\max}\) at the depot. However, when uncertainties are considered, a route satisfying constraint (A.4) may be infeasible due to the violation of deadlines at some customers. In addition, it may also violate the deadline at the depot. Constraints (A.5) are the generalized subtour elimination constraints (GSEC, Fischetti et al. (1998)), where \(S\) denotes any subset of customers along a route, and \(\delta(S)\) represents the set of arcs that connect customers in \(S\) with those outside \(S\). The master problem is solved in a branch-and-cut fashion and the GSEC and Benders cuts are added in fly.

After solving the master problem, we get its optimal solution \((\bar{x}, \bar{y})\) and extract \(K\) routes corresponding to \(\bar{x}_{ijk} = 1\).

For each route \(r_k = (0, v_1^k, v_2^k, \ldots, v_m^k, 0)\), the worst-case arrival time at each customer is computed via the recursion equation (16) or the equation (17). If route \(r_k\) is robust feasible,
then we compute the collected profit $\hat{\varrho}_k$ and generate the following Benders optimality cut:

$$\varrho_k \leq \hat{\varrho}_k + (\hat{\varrho}_k - M_{ub})W(x), \quad W(x) = \sum_{(i,j) \in r_k} x_{ijk} - y_{ik} - 1,$$

(A.8)

where $M_{ub}$ is an upper bound on the collected profit of each route, set as $M_{ub} = \sum_{i \in \mathcal{N}_c} p_i$.

Otherwise, there must exist at least one unreachable node whose arrival time exceeds its deadline. Let $\hat{r}_k = (v^k_1, \ldots, v^k_h)$ represent a sequence that contains the set of nodes from the first served customer $v^k_1$ to the first unreachable node $v^k_h$. Then, we generate the following Benders feasibility cut:

$$\sum_{i \in \{1, \ldots, h-1\}} x_{v^k_i v^k_{i+1}} \leq h - 2.$$  

(A.9)

**Appendix B  Statistics of Cuts Generated in the BBC Algorithm**

Table B1 provides statistical results of cuts generated in the BBC process, where #Optimality, #Feasibility, and #GSECs are the numbers of optimality, feasibility, and generalized subtour elimination cuts, respectively. #Total is the total number of cuts.

<table>
<thead>
<tr>
<th>$\Gamma$</th>
<th>#Optimality</th>
<th>#Feasibility</th>
<th>#GSECs</th>
<th>#Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>539</td>
<td>3779</td>
<td>1481</td>
<td>5799</td>
</tr>
<tr>
<td>1</td>
<td>407</td>
<td>5597</td>
<td>1551</td>
<td>7555</td>
</tr>
<tr>
<td>5</td>
<td>290</td>
<td>7229</td>
<td>1737</td>
<td>9256</td>
</tr>
<tr>
<td>10</td>
<td>271</td>
<td>7693</td>
<td>1812</td>
<td>9277</td>
</tr>
</tbody>
</table>