Transportation Research Part E 115 (2018) 246–266

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

A real-time adjustment strategy for the operational level stochastic orienteering problem: A simulation-aided optimization approach

Zheyong Bian, Xiang Liu*

Department of Civil and Environmental Engineering, Rutgers, The State University of New Jersey, 96 Frelinghuysen Road, Piscataway, NJ, 08854-8018, United States

ARTICLE INFO

Keywords:
Stochastic orienteering problem
Vehicle routing
Routing adjustment
Monte Carlo simulation
Multiple plan approach

ABSTRACT

This paper focuses on operational level stochastic orienteering problem, in which travel time and service time are stochastic and the vehicle can adjust its routing plan. A real-time adjustment strategy, called Simulation-Aided Multiple Plan Approach (SMPA), is proposed to optimize the real-time vehicle routing plan. We embed a "myopia prevention" strategy into SMPA to improve solution quality. The numerical experiment compares the performance of our proposed algorithm with a strategic level algorithm and another commonly used operational level algorithm called re-optimization algorithm. The results show that our algorithm outperforms previous methods in both solution quality and computing time.

1. Introduction

The orienteering problem (OP) is an extension of the classic traveling salesman problem (TSP). It is also known as the traveling salesman problem with profit (Feillet et al., 2005; Bérubé et al., 2009; Jozefowiez et al., 2008), the selective traveling salesman problem (Laporte and Martello, 1990; Gendreau et al., 1998a,b, Thomadsen and Stidsen, 2003) or the maximum collection problem (Kataoka and Morito, 1988; Butt and Cavalier, 1994; Butt and Ryan, 1999). The orienteering problem has a wide array of transportation and logistics applications, such as fuel delivery (Golden et al., 1987), single-ring design when building telecommunication networks (Thomadsen and Stidsen, 2003), tourist trip design (Vansteenwegen et al., 2007; Wörndl et al., 2017), mobile-crowdsourcing (Liao and Hsu, 2013; Chen et al., 2014), and unmanned aircraft and submarine surveillance activities (Wang et al., 2008; Evers et al., 2014a).

The OP can be defined as follows. A set of customers is given with a corresponding set of rewards. The terms "customer" and "reward" have different meanings in different applications. For example, in the Tourist Trip Design Problem (TTDP), "customers" is defined as a set of attractions to be visited in a city and the "reward" represents the interest level of an attraction. In unmanned aircraft activities, "customers" is a set of sites that are worth being surveilled and the "reward" is the value obtained if a site is surveilled. One vehicle should visit some customers selectively and determine a route with the objective of maximizing the total collected reward. The vehicle has a time budget, within which the vehicle must arrive at the pre-planned destination (Tsiligirides, 1984).

In practice, some elements of the orienteering problem are uncertain. For example, a truck's travel time on a link varies based on traffic conditions. Unmanned aircraft vehicles' travel time is influenced by wind, altitude and blocks. In addition, the service time for customers is not necessarily constant. Therefore, researchers proposed stochastic orienteering problems (SOP) for practical applications (Campbell et al., 2011; Tang and Miller-Hooks, 2005; Papapanagiotou et al., 2013, 2014, 2015a,b, 2016). This paper studies

* Corresponding author.
E-mail address: xiang.liu@rutgers.edu (X. Liu).

https://doi.org/10.1016/j.tre.2018.05.004
Received 1 October 2017; Received in revised form 6 April 2018; Accepted 11 May 2018

the **orienteering problem with stochastic travel and service time**. The objective of the stochastic orienteering problem is to maximize total reward while ensuring that the vehicle can arrive at the planned destination within the time budget for a probability greater than or equal to a required threshold (Varakantham et al., 2017).

On the operational level, some uncertain factors will be dynamically revealed after a vehicle visits certain customers, causing the *a priori* routing plan to possibly be no longer optimal for this operational-level stochastic orienteering problem (OSOP). For example, assume that a vehicle is executing its transportation and serving task based on a pre-planned route. After it visits some customers, the vehicle may not have enough time to finish its remaining tasks or the vehicle has redundant time to visit more customers if it continues to travel along the pre-planned route. At this time, the vehicle can adjust its route to ensure a high probability that the vehicle can reach the planned destination within the remaining time budget or to collect more rewards if the vehicle has a sufficient remaining time budget. Thus, this paper proposes a new computationally effective **operational-level adjustment strategy called simulation-aided multiple plan approach (SMPA)** to improve the vehicle routing plan in real time. Since the travel time and service time may not always follow tractable probabilistic distributions (e.g. normal distribution), it is difficult to analytically formulate the probability that the vehicle can reach the planned destination within a given time budget, which is called the "in-time arrival probability" (Varakantham et al., 2017). This paper applies the Monte Carlo simulation (Papapanagiotou et al., 2015b) to estimate the in-time arrival probability of a routing plan. The real-time adjustment strategy is implemented by the Multiple Plan Approach (MPA), which is inspired by a previous study by Bent and Van Hentenryck (2004). The basic idea of the MPA is to generate a solution pool, from which one solution is selected as the future routing plan that will be adopted by the vehicle to execute transportation and serving tasks. The solution pool is generated when the vehicle is traveling or serving a customer, and one solution is selected for future task execution each time the vehicle finishes serving a customer. The MPA can save decision making time when determining the next to-be-visited customer by fully utilizing the computer's leisure time to prepare a solution pool when the vehicle is traveling or serving a customer. However, simply using the MPA to adjust the routing plan in real time may cause the myopia problem, which specifically occurs in the OSOP. For example, simply adopting the optimal routing plan may sometimes cause regret for missing the opportunity to adopt a better routing plan. The myopia problem will be introduced in Section 4 in detail. To improve the quality of the routing plans, we incorporate a "myopia prevention" strategy into the SMPA. The "myopia prevention" strategy aims to minimize the expected regret reward when determining the next to-be-visited customer. Instead of simply adopting the optimal routing plan, the "myopia prevention" strategy preserves the probability to adopt the current routing plan in order not to lose the opportunity to adopt a potential routing plan with higher reward during the upcoming task execution process. Subsequently, a group of numerical examples are designed to test the proposed SMPA. The vehicle routing plan obtained by the real-time adjustment strategy SMPA will be compared with the *a priori* routing plan obtained by a strategic-level algorithm. Then another commonly used real-time adjustment strategy in the dynamic vehicle routing problem called re-optimization algorithm (Pillac et al., 2013) is employed to compare with the proposed SMPA in terms of the solution quality and computing time.

The remainder of the paper is organized as follows: Section 2 reviews existing literature on the orienteering problem. Section 3 introduces the operational-level stochastic orienteering problem. Section 4 proposes the real-time adjustment strategy called the simulation-aided multiple plan approach (SMPA). In section 5, numerical examples are designed and a simulation experiment is conducted to test the performance of the new algorithm compared to selected previous methods. Conclusions are drawn in Section 6 and future work is proposed in Section 7.

2. Literature review

2.1. Existing work

The orienteering problem was first proposed by Tsiligirides (1984) in the form of deterministic and static components. Since then, the problem has received wide attention because of the practical needs of transportation and logistics applications. Researchers have developed various algorithms that attempt to solve the orienteering problem (Leifer and Rosenwein, 1994; Wang et al., 1995; Fischetti et al., 1998; Tasgetiren and Smith, 2000; Schilde et al., 2009; Sevkli and Sevilgen, 2006; Liang et al., 2006; Campos et al., 2014; Chekuri and Kumar, 2004; Kobeaga et al., 2017; Ostrowski et al., 2017).

As attention to this problem has expanded, researchers have proposed some variations of orienteering problems, such as the orienteering problem with times window (OPTW) (Kantor and Rosenwein, 1992; Righini and Salani, 2006, 2009; Tricoire et al., 2010), the team orienteering problem (TOP) (Archetti et al., 2007; Chao, 1996a; Souffriau et al., 2010; Ke et al., 2008), capacitated team orienteering problem (Archetti et al., 2009; Tarantilis et al., 2013), the team orienteering problem with times window (TOPTW) (Vansteenwegen et al., 2009; Labadie et al., 2012; Gunawan et al., 2017), and the multi-objective orienteering problem (MOP) (Jozefowiez et al., 2008; Schilde et al., 2009). Some researchers applied different variants of OP into practice, such as tourist trip design (Baffo et al., 2015; De Falco et al., 2015), mobile-crowdsourcing (Liao and Hsu, 2013; Chen et al., 2014), and unmanned aircraft and submarine surveillance activities (Wang et al., 2008; Evers et al., 2014a). For more variants of the OP, please refer to Gunawan et al. (2016). However, all of these orienteering problems are deterministic and static. In practice, many elements of orienteering are uncertain and dynamic. Thus, more attention has recently been paid to the stochastic and dynamic orienteering problems.

Several researchers have studied stochastic orienteering problems. Ilhan et al. (2008) sought to solve the orienteering problem with *stochastic profits*. Evers et al. (2014b) assumed *stochastic weights* of the rewards collected by the vehicle in the orienteering problem. Zhang et al. (2016) assumed that the *customers' presence* was stochastic (i.e. each customer had a probability of requiring a visit) and optimized both the profit collected and the travel cost. Varakantham and Kumar (2013) and Varakantham et al. (2017)

considered *stochastic travel time* for the orienteering problem; they formulated a risk-aware stochastic orienteering problem as a scalable mixed integer linear programming problem. Gupta et al. (2012) assumed that the size of a node is random. In other words, they attempted to solve the orienteering problem with *stochastic service time*. Campbell et al. (2011), Tang and Miller-Hooks (2005), Papapanagiotou et al. (2013, 2014, 2015a,b, 2016) handled the orienteering problem with both stochastic service time and travel time.

Another variant of the orienteering problem is the dynamic orienteering problem, in which some elements, such as travel time and profit are updated as time progresses. Fomin and Lingas (2002), Li et al. (2010), Verbeeck et al. (2014a), and Garcia et al. (2010) considered time-dependent orienteering problems, in which travel time varies in different time periods. In their research, they assumed that the travel time can be accurately predicted, and thus the problem is essentially deterministic. Dynamic and stochastic orienteering problems have also been studied by several researchers. The orienteering problems studied by Lau et al. (2012), Verbeeck et al. (2014b), and Verbeeck (2016) have stochastic and time-dependent travel times. The travel time varies in different time stages and is stochastic with a known probability distribution.

2.2. Knowledge gaps

Although some researchers considered dynamic elements in the stochastic orienteering problem, they only focused on time-dependent travel time. Almost all previous studies on this subject focused on strategic-level optimization rather than on operational-level optimization. Their algorithms obtain an *a priori* routing plan in the planning stage. However, real-time adjustment strategies for operational-level optimization have not yet been well studied.

2.3. Intended contributions of this paper

To narrow the above-mentioned knowledge gaps, this paper aims to bring the following contributions to the body of knowledge:

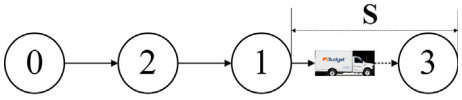
- To our knowledge, this paper is among the very few studies that focus on the operational-level stochastic orienteering problem (OSOP). In the OSOP, the vehicle can adjust its routing plan in real time when executing transportation and serving tasks so that the vehicle can collect more reward and ensure a higher in-time arrival probability.
- We propose a real-time adjustment strategy, called simulation-aided multiple plan approach (SMPA), to handle the OSOP with stochastic travel and service times that are subject to any general distributions (not necessarily analytical tractable distributions such as a normal distribution). The simulation technique is integrated in Multiple Plan Approach to direct the generation of solution pools and estimate the in-time arrival probabilities. The SMPA can save decision making time when determining the next to-be-visited customer by fully utilizing the computer's leisure time to obtain a solution pool for preparation when the vehicle is traveling or serving a customer.
- We embed a "myopia prevention" strategy into the SMPA to generate a routing plan that lets the vehicle collect more reward and simultaneously keep a high in-time arrival probability. The "myopia phenomenon" specifically occurs in the OSOP when a real-time adjustment strategy is used to solve OSOP. It minimizes the expected regret reward when determining the next to-be-visited customer. Instead of simply adopting the optimal routing plan, the "myopia prevention" strategy preserves the probability to adopt the current routing plan in order not to lose the opportunity to adopt a potential routing plan with higher reward during the upcoming task execution process.

2.4. The scope of this paper

This paper considers the operational-level stochastic orienteering problem, in which the travel times and service times are stochastic and the vehicle can adjust the routing plan in real time. We assume that the distributions of the travel times and service times are known in advance and are all time-independent. All customers are pre-determined and we do not consider the dynamically emerging customers when the vehicle is executing tasks. The rewards of all customers are deterministic and known.

3. Problem statement

Vansteenwegen et al. (2011) provided the basic description of the orienteering problem. Here, we define the orienteering problem with stochastic travel and service time. Let $\mathbf{CT} = \{1, 2, \ldots, n\}$ be a set of customers, which will be visited selectively. Let $\mathbf{DP} = \{0\}$ represent the departure location and $\mathbf{DT} = \{n + 1\}$ represent the destination of a vehicle. All the locations are denoted as the set $\mathbf{N} = \mathbf{CT} \cup \mathbf{DP} \cup \mathbf{DT}$. Associated with each $i \in \mathbf{CT}$, there is a reward r_i for the vehicle if it serves this customer. Let $\mathbf{r}$ be the collection of all rewards $\mathbf{r} = \{r_1, r_2, \ldots, r_n\}$. The time $t_{i,j}$ needed to travel from location i to j ($i, j \in \mathbf{N}$) and the service time st_i needed by customer i ($i \in \mathbf{CT}$) are assumed as random variables. Let $\mathbf{t} = \{t_{i,j} \mid i, j \in \mathbf{N}\}$ and $\mathbf{st} = \{st_i \mid i \in \mathbf{CT}\}$. Not all customers can be visited since the vehicle's available time is limited to a given time budget T. The objective of this problem is to maximize total collected reward with the constraint that the in-time arrival probability is greater than or equal to a threshold p. Thus, the stochastic orienteering problem (SOP) in this paper can be simply formulated by Formula (1). For a detailed formulation of the SOP, please refer to Varakantham et al. (2017).



S is represented by (S_1, S_2, S_3), where S_1 = "0, 2, 1", S_2 = 3, S_3 = **CT**\{2, 1, 3}

Fig. 1. An example of the current status **S**.

$$\max_X RW(X) \quad s.t. \quad IAP(X) \geqslant p \tag{1}$$

where X is a routing plan, $RW(X)$ obtains the total reward if the vehicle adopts the routing plan X, and $IAP(X)$ calculates the in-time arrival probability.

In the operational-level stochastic orienteering problem (OSOP), the routing plan adopted by the vehicle can be adjusted after the vehicle finishes serving certain customers. For example, after the vehicle finishes serving some customers, we may find that the vehicle has redundant time to serve more customers or that the vehicle is unable to serve all the planned customers within the remaining time budget. The vehicle can adjust the routing plan to earn more reward and reduce the risk that the vehicle cannot reach the planned destination within the time budget.

We use $\mathbf{S} = (S_1, S_2, S_3)$ to denote the status of the vehicle task execution, where S_1 is the sequence of served customers, S_2 is the next to-be-visited customer, and S_3 is the set of unvisited customers except S_2. Fig. 1 presents an example of the current status of **S**. In the figure, the vehicle is on the way to Customer 3. Thus, the sequence of visited customers S_1 = "0, 2, 1", the next planned to-be-visited customer S_2 = 3, and the set of unvisited customers S_3 = **CT**\{2, 1, 3}. The status **S** starts from the time when the vehicle departs from the last customer location (Customer 1 in Fig. 1) in S_1 and ends at the time when the vehicle finishes serving the next planned to-be-visited customer (Customer 3 in Fig. 1). At the end of status **S** (i.e. the vehicle finishes serving Customer 3 in Fig. 1), the task execution time when the vehicle finishes serving Customer S_2 is revealed, and the remaining time budget is known. The next to-be-visited customer should be determined and the current routing plan adopted by the vehicle may be adjusted based on the remaining time budget. This paper aims to propose such a real-time adjustment strategy with the objective of maximizing the collected reward with the constraint that the in-time arrival probability should be greater than or equal to a certain threshold.

4. Simulation-aided multiple plan approach (SMPA)

4.1. Overview of the SMPA

This section introduces a new algorithm, called simulation-aided multiple plan approach (SMPA), to solve the OSOP. For notations, except when specified, please refer to Table 1. This table only defines the most important notations and operators that are commonly used in all algorithms. We specify the meaning of all newly-defined notations in the algorithms in Appendix A.

This subsection gives a brief overview of the SMPA. The Multiple Plan Approach (MPA) was originally proposed by Bent and Van Hentenryck (2004). The basic idea of the MPA is to utilize the computer's leisure time when the vehicle is traveling or serving customers to generate multiple solutions preparing for determination of a future routing plan. This paper proposes a simulation-aided MPA with the "myopia prevention" strategy (explained later) to solve the OSOP.

In order to clarify the computing activities of the SMPA, we define three periods of the vehicle's activities, 1) preparing period, 2) traveling and serving period, and 3) decision making period. We will give the definitions of the three periods and clarify the results that will be obtained in the three periods below. Fig. 2 shows the computer activities and vehicle activities based on the time and spatial horizons.

1) The preparing period ($\mathbf{S} = (S_1 = \varnothing, S_2 = 0, S_3 = \mathbf{CT})$) is the period before the vehicle departs from the departure location. In this period, an *a priori* routing plan should be determined to direct the vehicle task execution. Initially, the vehicle will execute the task based on the *a priori* solution.
2) Traveling and serving periods (during a status **S**). After the preparing period finishes, the vehicle starts to execute transporting and serving tasks. When the vehicle is on the way or serving a customer, a solution pool should be generated to prepare for the determination of the next to-be-visited customer in the decision making periods.
3) Decision making periods (at the end of a status **S**): when the vehicle finishes serving each customer, the remaining time budget is revealed, and the next to-be-visited customer should be determined and the routing plan adopted by the vehicle may be updated.

The three periods have different requirements for the promptness of computational process. The preparing period usually has the lowest requirement because usually there is enough time to determine an *a priori* solution before the vehicle starts to execute the tasks. The traveling and serving periods do not require that the algorithm should make any decision, but requires the algorithm to obtain the solution pool that will be used in the decision making period. The decision making periods have the highest requirement for the computing promptness. This is because when the vehicle finishes serving a customer, a prompt decision should be made to determine which customer will be visited and served next. If the algorithm needs to spend a very long time in determining the next

customer to be served, most of the time budget will be used for computing instead of task execution.

The proposed SMPA reasonably allocates the computing time into three periods based on the vehicle activities. Fig. 3 summarizes the algorithms when the vehicle is executing transportation and serving tasks. When the vehicle is waiting at the departure location, the computer uses **Algorithm 2** to generate an initial solution pool. Then, **Algorithm 4** is used to determine the initial routing plan (the *a priori* routing plan) adopted by the vehicle. After the vehicle departs from the departure location and when the vehicle is on the way or serving a customer, **Algorithm 2** generates a new solution pool and **Algorithm 3** transfers compatible solutions (the definition of "compatible solutions" will be introduced in Subsection 4.4) into the current solution pool. When the vehicle finishes serving a customer, the next to-be-visited customer will be determined by **Algorithms 4 and 5**. **Algorithm 4** is used to find an optimal alternative routing plan. Then **Algorithm 5** determines if the optimal alternative solution replaces the current routing plan based on the "myopia" prevention strategy to direct the vehicle task execution. Based on the updated routing plan, the vehicle departs to the next customer. The computer conducts the same computing activities in the next cycle until the time budget runs out. Note that both Algorithms 4 and 5 need to estimate the in-time arrival probability and thus **Algorithm 1** (Monte Carlo simulation) is embedded into Algorithms 4 and 5 in all phases.

Table 1
Notations and operators.

Notations	
$\mathbf{CT}$	Set of customers, $\mathbf{CT} = \{1, 2, \ldots, n\}$.
$\mathbf{DP}$	The departure location, $\mathbf{DP} = \{0\}$.
$\mathbf{DT}$	The destination, $\mathbf{DT} = \{n + 1\}$.
$\mathbf{N}$	Set of all locations, including the customers, the departure location and the destination, $\mathbf{N} = \mathbf{CT} \cup \mathbf{DP} \cup \mathbf{DT}$.
$\mathbf{r}$	Collection of all rewards $\mathbf{r} = \{r_1, r_2, \ldots, r_n\}$.
$t_{i,j}$	Travel time from location i to location j ($i, j \in \mathbf{N}$). $t_{i,j}$ is a stochastic variable, whose distribution is known.
st_i	Service time on customer i ($i \in \mathbf{CT}$). It is a stochastic variable and the distribution is known.
$rt_{i,j}$	Real travel time from location i to location j ($i, j \in \mathbf{N}$).
rst_i	Real service time on customer i ($i \in \mathbf{CT}$).
p	The minimum required probability that the vehicle can reach the planned destination within the time budget.
T	Total time budget for the vehicle to execute tasks.
$\mathbf{S}$	The current status of the vehicle's task execution, $\mathbf{S} = (S_1, S_2, S_3)$. S_1 is the routing sequence of the served customers, S_2 is the next to-be-visited customer, and S_3 is the set of unvisited customers except S_2. This status lasts from the time when the vehicle finishes serving the last customer in S_1 to the time when the vehicle finishes serving S_2.
$IAP^{\mathbf{S}}(X)$	The in-time arrival probability of the routing plan X given that the travel times and service times are revealed at the end of status $\mathbf{S}$.
$\mathbf{X}^{\mathbf{S}}$	The solution pool generated in the current status $\mathbf{S}$, $\mathbf{X}^{\mathbf{S}} = \{X_i^{\mathbf{S}}, i = 1, 2, 3, \ldots\}$. If $X_i^{\mathbf{S}}$ is one of the solutions, the routing plan $X_i^{\mathbf{S}}$ can be represented by "S_1, S_2, $R^{\mathbf{S}}(X_i^{\mathbf{S}})$" where "$S_1$" is the routing sequence of served customers, "S_2" is the next to-be-visited customer and $R^{\mathbf{S}}(X_i^{\mathbf{S}})$ is the routing sequence that the vehicle has not started to travel along yet.
$X^{\mathbf{S}}$	The solution with the maximum reward among the solutions in the solution pool $\mathbf{X}^{\mathbf{S}}$ whose in-time arrival probabilities are greater than or equal to p. It can also be treated as the optimal solution in $\mathbf{X}^{\mathbf{S}}$, which is obtained by Algorithm 4.
$X_c^{\mathbf{S}}$	The current routing plan adopted by the vehicle in the status $\mathbf{S}$.
$\mathbf{XP}^{\mathbf{S}}$	The preliminary screened solution pool. It is the solution pool obtained by Algorithm 4, removing solutions, whose in-time arrival probabilities are obviously smaller than p, from the solution pool $\mathbf{X}^{\mathbf{S}}$.
$t^{\mathbf{S}}$	At the end of status $\mathbf{S}$, the real travel time from $(S_1)_{end}$ to S_2 and the service time on customer S_2 are revealed. $t^{\mathbf{S}}$ is the summation of the travel time from $(S_1)_{end}$ to S_2 and the service time on customer S_2, where $(S_1)_{end}$ is the last customer in the routing sequence S_1 (please see the operators below).
$T_b^{\mathbf{S}}$	The remaining time budget at the beginning of the status $\mathbf{S}$.
$T_e^{\mathbf{S}}$	The remaining time budget at the end of the status $\mathbf{S}$.
EI	The expected potential increased reward if $X_c^{\mathbf{S}}$ is still adopted as the current routing plan at the end of the status $\mathbf{S}$.
ED	The expected potential decreased reward if $X_c^{\mathbf{S}}$ is still adopted as the current routing plan at the end of the status $\mathbf{S}$.
$pd(XP_{ic}^{\mathbf{S}})/pd(X_c^{\mathbf{S}})$	$pd(XP_{ic}^{\mathbf{S}})$ is the probability that $XP_{ic}^{\mathbf{S}}$'s in-time arrival probability can reach the value greater than or equal to the threshold p in the future when the vehicle departs to the first customer in $XP_{ic}^{\mathbf{S}}$ which is different from that in $X_c^{\mathbf{S}}$. $pd(X_c^{\mathbf{S}})$ is the probability that $X_c^{\mathbf{S}}$'s in-time arrival probability can reach the value greater than or equal to the threshold p when the vehicle departs to the first customer in $X_c^{\mathbf{S}}$ different from that in $XP_{ic}^{\mathbf{S}}$.
Operators	
$(X)_i$	Obtain the ith customer in the routing sequence X.
$(X)_{end}$	Obtain the last customer in the routing sequence X.
$(X)_{i:j}$	Obtain the routing sequence from ith location to the jth location in the routing sequence X.
"X, a"	X is a routing sequence. "X, a" is an operator that puts Customer a behind the routing sequence X and forms a new routing sequence.
$L(X)$	Obtain all connected links in the routing sequence X.
$C(X)$	Obtain all customers in the routing sequence X.
$RW(X)$	X is a routing plan. $RW(X)$ obtains the total reward of the routing plan.
$\boldsymbol{RW}(\mathbf{X})$	$\mathbf{X}$ is a solution pool, $\boldsymbol{RW}(\mathbf{X})$ obtains the rewards of all the routing plans in $\mathbf{X}$.
$R^{\mathbf{S}}(X)$	X is a full routing plan with departure location, customer locations and the destination. In the status $\mathbf{S}$, $R^{\mathbf{S}}(X)$ is the routing sequence behind S_2 in the full routing plan X. For example, if X is "1–3–4–5–1", when the vehicle is in the link "3-4" or serving customer 4, $R^{\mathbf{S}}(X)$ is the routing sequence "5-1".
$U^{\mathbf{S}}(X)$	$U^{\mathbf{S}}(X)$ = "S_2, $R^{\mathbf{S}}(X)$", For example, if X is "1-3-4-5-1", when the vehicle is in the link "3-4" or serving customer 4, $U^{\mathbf{S}}(X)$ is the routing sequence "4-5-1".

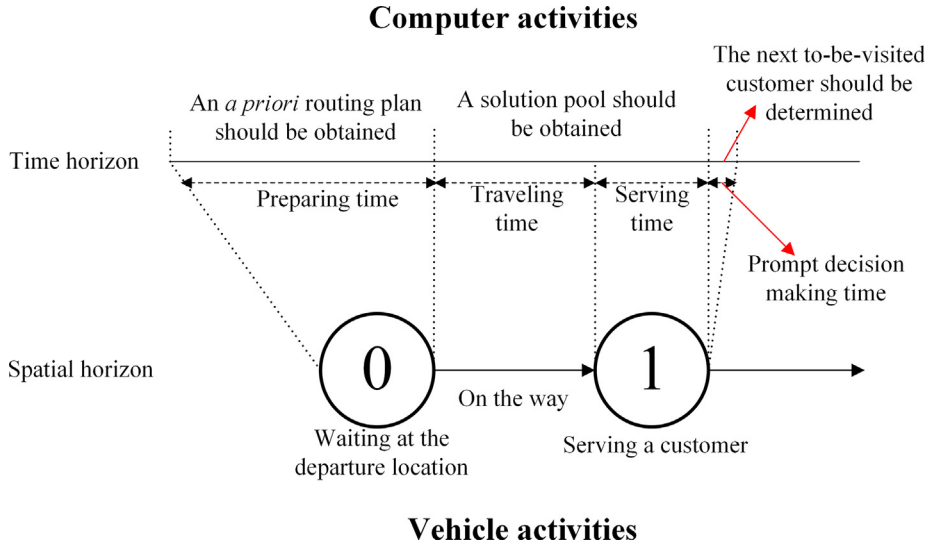


Fig. 2. Computing activities and vehicle activities.

4.2. Monte Carlo simulation to estimate the in-time arrival probability

Each time the vehicle finishes serving a customer, the real-time adjustment strategy should determine a future routing plan that can direct the vehicle to execute the tasks. The in-time arrival probabilities of routing plans are required to be estimated in each decision making period. It is difficult to measure the in-time arrival probability of a routing plan analytically if the travel times and service times do not follow tractable distributions (e.g. normal distributions). Papapanagiotou et al. (2014, 2015a) used the Monte Carlo simulation approach to measure the objective function of the model for their stochastic orienteering problem. This paper uses Monte Carlo simulation to measure the in-time arrival probabilities.

At the end of the status **S** when the vehicle finishes serving customer S_2, the task execution time of the routing sequence "S_1, S_2", including all travel times and service times, are revealed, and thus the in-time arrival probability of a routing plan X can be estimated based on the remaining routing sequence $R^{\mathbf{S}}(X)$ and the remaining time budget $T_e^{\mathbf{S}}$. We use Formula (2) to represent the inputs and outputs of the Monte Carlo simulation. For the pseudocode, please refer to Algorithm 1 in Appendix A.

$$IAP^{\mathbf{S}}(X) = \textbf{Algorithm 1}\ (X, T_e^{\mathbf{S}}, \mathbf{t}, \mathbf{st}, NSR) \tag{2}$$

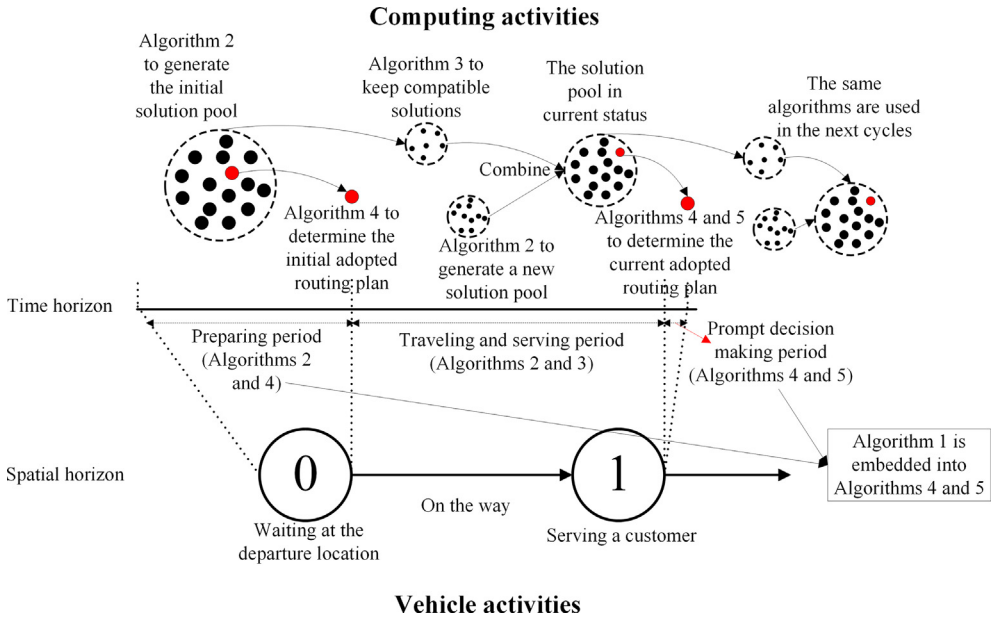


Fig. 3. Summary of algorithms used in different periods.

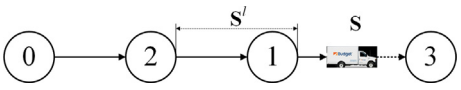


Fig. 4. Relationship between statuses **S** and $\mathbf{S}^l$.

The inputs include:

X: the solution that needs to be estimated;
$T_e^{\mathbf{S}}$: remaining time budget for the vehicle to execute tasks at the end of status **S**;
t: collection of all travel times between the two locations $\mathbf{t} = \{t_{i,j} \mid i, j \in \mathbf{N}\}$;
st: service times for all customers $\mathbf{st} = \{st_i \mid i \in \mathbf{CT}\}$;
NSR: the number of simulation replications.

The output is

$IAP^{\mathbf{S}}(X)$: estimated in-time arrival probability of solution X at the end of status **S**.

4.3. Solution generation algorithm

This algorithm is used in both of the preparing period and the traveling and serving periods. In the preparing period, this algorithm is responsible for generating an initial solution pool; in the traveling and serving periods, this algorithm is used to generate a new solution pool. We use Tabu Search (TS) to implement this algorithm. TS can avoid the repeated generation of identical solutions using a memory function (Gendreau et al., 1998b). In each iteration of TS, a group of routing plans are selected to be put into the solution pool. In TS, travel times and service times are randomly and periodically generated to direct the iteration process so that various solutions are generated in the solution pool. An objective function is proposed to measure the quality of a solution ($X_i^{\mathbf{S}}$) in the current status **S** based on the generated travel times and service times.

$$Z(X_i^{\mathbf{S}}) = \sum_{i \in C(X_i^{\mathbf{S}})} r_i - M \cdot \max\left(\sum_{i,j \in L(U^{\mathbf{S}}(X_i^{\mathbf{S}}))} t_{i,j}^{icp} + \sum_{i \in C(R^{\mathbf{S}}(X_i^{\mathbf{S}}))} st_i^{icp} - T_b^{\mathbf{S}}, 0 \right) - m \cdot \min\left(\sum_{i,j \in L(U^{\mathbf{S}}(X_i^{\mathbf{S}}))} t_{i,j}^{icp} + \sum_{i \in C(R^{\mathbf{S}}(X_i^{\mathbf{S}}))} st_i^{icp} - T_b^{\mathbf{S}}, 0 \right) \quad (3)$$

We divide the iterations of TS into *NP* computing periods. In each computing period, the travel times and service times are re-simulated by the computer to direct the iteration process of TS so that various solutions can be generated. In the *icp*th computing period of TS, $t_{i,j}^{icp}$ is the travel time from location i to j and st_i^{icp} is the service time for Customer i. Note that all $t_{i,j}^{icp}$ and st_i^{icp} are simulated by the computer to conduct the search process of TS and their values are not real. M is a number that is much greater than any r_i ($i \in \mathbf{CT}$), and m is a number that is much smaller than any r_i ($i \in \mathbf{CT}$) but is greater than "0". When the total task execution time exceeds the remaining time budget, the solution will be given a very low objective function value. This mechanism is achieved by the second term in Formula (3). If the total task execution times of two routing plans do not exceed the time budget and the two routing plans have identical rewards collected, the routing plan with less task execution time will be favored. This mechanism is achieved by the third term in Formula (3).

We use Formula (4) to represent this algorithm. For the detailed pseudocode, please refer to Algorithm 2 in Appendix A.

$$\mathbf{X}^{\mathbf{S}} = \textbf{Algorithm } 2\,(T_b^{\mathbf{S}}, \mathbf{t}, \mathbf{st}) \quad (4)$$

The inputs include:

$T_b^{\mathbf{S}}$: the remaining time budget when the status **S** starts;
t: collection of all travel times between two locations $\mathbf{t} = \{t_{i,j} \mid i, j \in \mathbf{N}\}$;
st: service times for all customers $\mathbf{st} = \{st_i \mid i \in \mathbf{CT}\}$.

The output is

$\mathbf{X}^{\mathbf{S}}$: a solution pool generated in status **S**.

4.4. Compatible solution transfer algorithm

This algorithm is used in the traveling and serving periods and aims to keep the compatible solutions from the solution pool in last status to the solution pool in current status. This algorithm can save the computing time of Algorithm 2 by reducing the number of solutions generated in the current status **S**. Let $\mathbf{S}^l = (S_1{}^l, S_2{}^l, S_3{}^l)$ denote the last status of the vehicle task execution. Fig. 4 shows the relationship between **S** and $\mathbf{S}^l$. Let $\mathbf{X}^{\mathbf{S}^l} = \{X_i^{\mathbf{S}^l}, i = 1,2,3,\ldots\}$ denote the solution pool generated in the last status $\mathbf{S}^l$. If $S_2 = (R^{\mathbf{S}^l}(X_i^{\mathbf{S}^l}))_1$, $X_i^{\mathbf{S}^l}$ is a compatible solution in the current status **S**, otherwise it is an incompatible solution. We use Fig. 5 to introduce the concept of the compatible solutions and incompatible solutions. In Fig. 5, the next to-be-visited customer (Customer 3) is already determined, and

the vehicle is now traveling from Customer 1 to Customer 3. Solutions 1 and 2 are two solutions from the previous solution pool generated in the last status before the vehicle finishes serving Customer 1. Since the Customer 3 is already determined as the next to-be-visited customer, Solution 2, in which the next to-be-visited customer is Customer 5, is no longer valid. Thus, Solution 2 is defined as an incompatible routing plan in the current status. In Solution 1, Customer 3 is the next to-be-visited customer. It is compatible with the current status of task execution and thus Solution 1 is defined as a compatible routing plan in the current status.

The "compatible solution transfer algorithm" can be expressed as Formula (5) and the pseudocode is presented in Appendix A.

$$\mathbf{X^S} = \textbf{Algorithm } 3\ (\mathbf{X^S}, \mathbf{X^{S^l}}) \tag{5}$$

The inputs include:

$\mathbf{X^S}$: the new solution pool in status **S** generated by Algorithm 2;
$\mathbf{X^{S^l}}$: the solution pool in last status **S***l*.

The output is

$\mathbf{X^S}$: the combined solution pool, including the transferred solutions and the newly generated solutions.

4.5. Solution selection algorithm

Fig. 3 shows that Algorithm 4 is used in the preparing period and the decision making periods. When the algorithm is used in the preparing period, it determines an initial *a priori* routing plan that is adopted by the vehicle. When it is used in the decision making periods, it obtains a candidate routing plan that may or may not replace the current adopted routing plan according to Algorithm 5. At the end of status **S**, the vehicle finishes serving the customer S_2. The travel time from $(S_1)_{end}$ to S_2 plus the service time on customer S_2 is revealed, which is denoted as $t^{\mathbf{S}}$, $t^{\mathbf{S}} = rt_{(S_1)_{end},S_2} + rst_{S_2}$. Thus, the remaining time budget at the end of status **S** is $T_e^{\mathbf{S}} = T_b^{\mathbf{S}} - t^{\mathbf{S}}$.

We need to select a solution $X^{\mathbf{S}}$ from the solution pool ($\mathbf{X^S}$). This solution ($X^{\mathbf{S}}$) earns the maximum reward among all the solutions in $\mathbf{X^S}$ while the in-time arrival probability should be greater than or equal to the given value p.

$$X^{\mathbf{S}} = \arg\max_j \{RW(X_j^{\mathbf{S}}) | X_j^{\mathbf{S}} \in \mathbf{X^S}, IAP^{\mathbf{S}}(X_j^{\mathbf{S}}) \geqslant p\} \tag{6}$$

The in-time arrival probability is measured by Monte Carlo simulation (Algorithm 1). In order to save computing time, the algorithm firstly uses a preliminary Monte Carlo simulation to remove the routing plans whose task completion times obviously exceed the remaining time budget. In the preliminary Monte Carlo simulation, the number of simulation replications is relatively small. We assume that $\mathbf{XP^S}$ is the solution pool after a preliminary screening. Then, $X^{\mathbf{S}}$ is selected from the solution pool $\mathbf{XP^S}$ by increasing the number of replications in the Monte Carlo simulation. The solution selection algorithm can be expressed as Formula (7) and the pseudocode is given in Appendix A.

$$(X^{\mathbf{S}}, \mathbf{XP^S}, T_e^{\mathbf{S}}) = \textbf{Algorithm } 4\ (\mathbf{X^S}, T_b^{\mathbf{S}}, t^{\mathbf{S}}, p, \mathbf{t}, \mathbf{st}) \tag{7}$$

The inputs include:

$\mathbf{X^S}$: the solution pool $\{X_1^{\mathbf{S}}, X_2^{\mathbf{S}}, \ldots\}$ in the status **S**;
$T_b^{\mathbf{S}}$: the remaining time budget when the status **S** starts;
$t^{\mathbf{S}}$: the real travel time from $(S_1)_{end}$ to S_2 plus the service time on customer S_2;
p: the required minimum in-time arrival probability;
t: collection of all travel times between two locations $\mathbf{t} = \{t_{i,j} \mid i, j \in \mathbf{N}\}$;
st: service times for all customers $\mathbf{st} = \{st_i \mid i \in \mathbf{CT}\}$.

The outputs include:

$X^{\mathbf{S}}$: the optimal solution selected from $\mathbf{X^S}$ in status **S**;
$\mathbf{XP^S}$: preliminary screened solution pool in status **S**.
$T_e^{\mathbf{S}}$: the remaining time budget when the status **S** ends.

4.6. Algorithm for updating the current routing plan adopted by the vehicle

The routing plan adopted by the vehicle in the status **S** is $X_c^{\mathbf{S}}$. Initially, the vehicle executes the tasks based on the *a priori* solution determined in the preparing period. When the vehicle starts to execute the tasks, the current routing plan $X_c^{\mathbf{S}}$ may be updated after the vehicle finishes serving some customers. Based on Algorithm 4, a solution $X^{\mathbf{S}}$ is selected from the solution pool $\mathbf{X^S}$ in the status **S**. This solution has the maximum reward among the solutions in $\mathbf{X^S}$ whose in-time arrival probability is greater than or equal to p. In other words, it is the optimal solution in $\mathbf{X^S}$. Intuition tells us that the vehicle should adopt $X^{\mathbf{S}}$ as the current routing plan instead of $X_c^{\mathbf{S}}$. However, simply replacing $X_c^{\mathbf{S}}$ with $X^{\mathbf{S}}$ sometimes causes the "myopia" problem.

Let us use the example in Fig. 6 to demonstrate the myopia problem. We assume that the in-time arrival probability should be

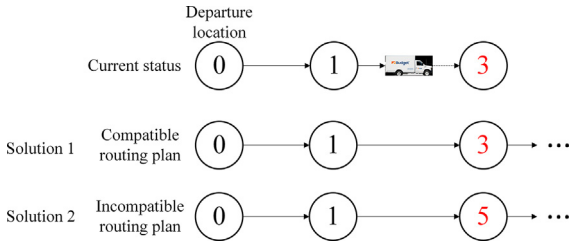


Fig. 5. Examples of a compatible solution and an incompatible solution.

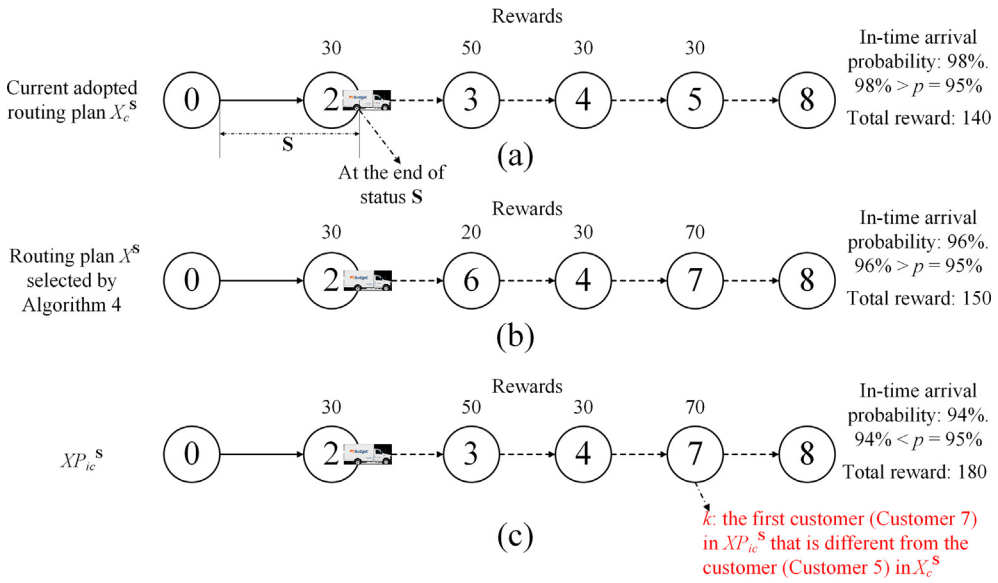


Fig. 6. An example demonstrating the myopia problem.

greater than or equal to the prescribed minimum threshold 95% in the example. In Fig. 6(a), the reward of the current adopted routing plan X_c^S is "140". After customer 1 is served, the in-time arrival probability of the current adopted routing plan is 98%. Algorithm 4 finds a better routing plan X^S (Fig. 6(b)) whose reward is "150" and whose in-time arrival probability is 96%. However, if X^S replaces X_c^S as the current adopted routing plan, other potential routing plans which are incompatible with X^S will never be selected as the current adopted routing plan. For example, Fig. 6(c) is a routing plan with reward "180", which is compatible with X_c^S but incompatible with X^S. Although the in-time arrival probability of the routing plan in Fig. 6(c), 94%, is smaller than the prescribed minimum threshold 95%, it is possible that the in-time arrival probability will be greater than or equal to 95% after the vehicle finishes serving more customers. If X^S replaces the current routing plan X_c^S, the routing plan in Fig. 6(c) will never be adopted by the vehicle and thus the vehicle misses the opportunity to increase the reward to "180". If the current routing plan X_c^S does not change, it is possible that the routing plan in Fig. 6(c) with greater reward (180) will be adopted by the vehicle in the future after finishing serving Customer 4. This situation is the "myopia" problem when X^S simply replaces the current routing plan X_c^S.

In order to avoid the "myopia" problem, we adopt the following "myopia prevention" strategy, which can improve the expected reward of a routing plan. At the end of the status **S**, there are three possible situations.

1) The next customers to be visited in solution X^S and X_c^S are identical.
2) The next customers to be visited in solution X^S and X_c^S are different and the reward of X^S is greater than that of X_c^S.
3) The next customers to be visited in solution X^S and X_c^S are different, and the reward of X^S is smaller than that of X_c^S because the in-time arrival probability of X_c^S at the end of the status **S** is smaller than the required threshold p.

If the first situation happens, we keep X_c^S as the current routing plan. In situation 2, we need to judge if the increased reward ($RW(X^S) - RW(X_c^S)$) of X^S can compensate for the expected potential increased reward (EI) if the X_c^S is adopted as the current routing plan. Now let us define the expected potential increased reward (EI) if the X_c^S is adopted as the current routing plan. EI can also be treated as the expected "more reward" that the vehicle loses the opportunity to earn if X^S replaces X_c^S.We need to find if there is at least one solution XP_{ic}^S in the preliminary screened solution pool $\mathbf{XP}^S$ that $RW(XP_{ic}^S) > RW(X^S)$ and the next to-be-visited customer in XP_{ic}^S is identical with that in X_c^S, i.e. $(R^S(X_c^S))_1 = (R^S(XP_{ic}^S))_1$. In fact, XP_{ic}^S is a potential routing plan with higher reward that the vehicle may adopt in the future if the X_c^S is still adopted as the current routing plan. If no such solution XP_{ic}^S exists, the solution X^S is

Table 2
Algorithm to obtain $pd(XP_{ic}^{\mathbf{S}})$.

Algorithm to obtain $pd(XP_{ic}^{\mathbf{S}})$

$C\_h = 0$;
$C\_l = 0$;
For $h = 1 : H$
  Use Monte Carlo to get hth simulation of tb as tb_h.
  For $l = 1 : L$
   Use Monte Carlo to get lth simulation of ta as ta_l.
   If $tb_h + ta_l \leqslant T_e^{\mathbf{S}}$
   $C\_l = C\_l + 1$;
   End if
  End for
  If $\frac{C\_l}{L} \geqslant p$
   $C\_h = C\_h + 1$;
  End if
End for
$pd(XP_{ic}^{\mathbf{S}}) = \frac{C\_h}{H}$

accepted as the current routing plan. If such solution $XP_{ic}^{\mathbf{S}}$ exists, we should estimate the probability ($pd(XP_{ic}^{\mathbf{S}})$) that $XP_{ic}^{\mathbf{S}}$ is qualified to replace $X_c^{\mathbf{S}}$ for the vehicle to execute tasks in the future. In other words, $pd(XP_{ic}^{\mathbf{S}})$ is the probability that $XP_{ic}^{\mathbf{S}}$'s in-time arrival probability can reach the value greater than or equal to the threshold p in the future when the vehicle starts to depart to the first customer in $XP_{ic}^{\mathbf{S}}$ that is different from that in $X_c^{\mathbf{S}}$.

We use Fig. 6 to demonstrate how this probability is defined. The first two customers (Customers 3 and 4) after status **S** in $X_c^{\mathbf{S}}$ are identical with those in $XP_{ic}^{\mathbf{S}}$. Customer 5 in $X_c^{\mathbf{S}}$ is the first customer which is different from that (Customer 7) in $XP_{ic}^{\mathbf{S}}$. Thus, $pd(XP_{ic}^{\mathbf{S}})$ is the probability that the in-time arrival probability increases from "94%" in current status to "≥95%" when the vehicle finishes serving Customer 4 before departing to Customer 7 in Fig. 6(c). At the time when the vehicle finishes serving Customer 4, two layers of uncertainty should be considered to calculate $pd(XP_{ic}^{\mathbf{S}})$.

1) The first layer of uncertainty is whether the vehicle can arrive at the planned destination in time. This uncertainty introduces the in-time arrival probability when the vehicle finishes serving Customer 4. This uncertainty results from the stochastic time of $ta = t_{4,7} + t_{7,8} + st_7$ in Fig. 6(c).
2) The second layer of uncertainty is whether the in-time arrival probability is greater than or equal to the threshold p when the vehicle finishes serving Customer 4. This uncertainty is represented by $pd(XP_{ic}^{\mathbf{S}})$. This uncertainty is caused by the stochastic remaining time budget when the vehicle finishes serving Customer 4. The remaining time budget is stochastic because of the stochastic time of $tb = t_{2,3} + t_{3,4} + st_3 + st_4$ in Fig. 6(c). Therefore, this indicates that the in-time arrival probability is conditional on the time of tb ($tb = t_{2,3} + t_{3,4} + st_3 + st_4$).

Based on the two layers of the uncertainty, $pd(XP_{ic}^{\mathbf{S}})$ in Fig. 6(c) can be formulated as:

$$pd(XP_{ic}^{\mathbf{S}}) = P(P(tb + t_{4,7} + t_{7,8} + st_7 \leqslant T_e^{\mathbf{S}} | tb = t_{2,3} + t_{3,4} + st_3 + st_4) \geqslant p).$$

pd is obtained in the following way. Let $ta = t_{4,7} + t_{7,8} + st_7$ and $tb = t_{2,3} + t_{3,4} + st_3 + st_4$. Both ta and tb are unknown since both of the two parts of the tour are not traversed yet. Thus, simulation is needed to get possible values of ta and tb. We use Monte Carlo to simulate $tb = t_{2,3} + t_{3,4} + st_3 + st_4$ for H replications. We denote the hth simulation of tb as tb_h, $h = 1, 2, \ldots, H$. For each specific simulation result of tb_h, we also use Monte Carlo to simulate $ta = t_{4,7} + t_{7,8} + st_7$ for L replications. We denote the lth simulation of ta as ta_l, $l = 1, 2, \ldots, L$. pd is calculated by the algorithm in Table 2.

If $X_c^{\mathbf{S}}$ is still adopted as the current routing plan, the expected increase in reward is defined as

$$EI = \max_{XP_{ic}^{\mathbf{S}} \in \mathbf{XP}^{\mathbf{S}}} pd(XP_{ic}^{\mathbf{S}}) \times (RW(XP_{ic}^{\mathbf{S}}) - RW(X_c^{\mathbf{S}})) \tag{8a}$$

Subject to

$$RW(XP_{ic}^{\mathbf{S}}) > RW(X^{\mathbf{S}}) \tag{8b}$$

$$(R^{\mathbf{S}}(X_c^{\mathbf{S}}))_1 = (R^{\mathbf{S}}(XP_{ic}^{\mathbf{S}}))_1 \tag{8c}$$

$$pd(XP_{ic}^{\mathbf{S}}) = P(P(tb^k(XP_{ic}^{\mathbf{S}}) + ta^k(XP_{ic}^{\mathbf{S}}) \leqslant T_e^{\mathbf{S}} \mid tb^k(XP_{ic}^{\mathbf{S}})) \geqslant p) \tag{8d}$$

where k is the minimum number that $(R^{\mathbf{S}}(XP_{ic}^{\mathbf{S}}))_k \neq (R^{\mathbf{S}}(X_c^{\mathbf{S}}))_k$ (in Fig. 6, $(R^{\mathbf{S}}(X_c^{\mathbf{S}}))_k = 5$, $(R^{\mathbf{S}}(XP_{ic}^{\mathbf{S}}))_k = 7$), $tb^k(XP_{ic}^{\mathbf{S}})$ is the task execution time of the routing sequence before departing to the kth customer (Customer 7 in Fig. 6(c)) in $XP_{ic}^{\mathbf{S}}$, and $ta^k(XP_{ic}^{\mathbf{S}})$ is the task execution time of the routing sequence after the vehicle finishes serving the $(k-1)$th customer. Formula (8a) is to obtain the maximum

expected increased reward of all solutions in $\mathbf{XP^S}$. Formula (8b) ensures that the solutions in $\mathbf{XP^S}$ with the reward smaller than the reward of X^S have no opportunity to calculate *EI*. Formula (8c) ensures that the next to-be-visited customer in the solution $XP_{ic}^{\mathbf{S}}$ should be identical with that in solution $X_c^{\mathbf{S}}$, otherwise $XP_{ic}^{\mathbf{S}}$ has no opportunity to calculate *EI*. Formula (8d) is to calculate the probability that $XP_{ic}^{\mathbf{S}}$ is qualified to replace $X_c^{\mathbf{S}}$ for the vehicle to execute tasks in the future. If $EI < (RW(X^{\mathbf{S}})\text{-}RW(X_c^{\mathbf{S}}))$, we replace $X_c^{\mathbf{S}}$ with $X^{\mathbf{S}}$ as the current routing plan. Otherwise, $X_c^{\mathbf{S}}$ is kept as the current routing plan. For the example in Fig. 6, assuming that $pd(XP_{ic}^{\mathbf{S}}) = 0.4$, $EI = 0.4 \times (180\text{–}140) = 16 > RW(X^{\mathbf{S}}) - RW(X_c^{\mathbf{S}}) = 150\text{–}140 = 10$. Thus $X_c^{\mathbf{S}}$ is kept as the current routing plan.

In situation 3, we need to judge if the decreased reward $(RW(X_c^{\mathbf{S}}) - RW(X^{\mathbf{S}}))$ of $X^{\mathbf{S}}$ is smaller than the expected potential decreased reward (*ED*) if $X_c^{\mathbf{S}}$ is still adopted as the current routing plan. We need to find if there is at least one solution $XP_{ic}^{\mathbf{S}}$ in the preliminary screened solution pool $\mathbf{XP^S}$ that $IAP^{\mathbf{S}}(XP_{ic}^{\mathbf{S}}) \geq p$ and the next to-be-visited customer in $XP_{ic}^{\mathbf{S}}$ is identical with that in $X_c^{\mathbf{S}}$, i.e.$(R^{\mathbf{S}}(X_c^{\mathbf{S}}))_1 = (R^{\mathbf{S}}(XP_{ic}^{\mathbf{S}}))_1$. In fact, $XP_{ic}^{\mathbf{S}}$ is an alternative routing plan with smaller reward that the vehicle can adopt in the future after the $X_c^{\mathbf{S}}$ is still adopted as the current routing plan.

If such solution $XP_{ic}^{\mathbf{S}}$ does not exist, $X^{\mathbf{S}}$ will be accepted as the current routing plan. This is because if there is no such alternative routing plan $XP_{ic}^{\mathbf{S}}$ in $\mathbf{XP^S}$ with smaller reward, it is likely for the vehicle to use solutions with even much lower rewards. Otherwise, the in-time arrival probability may be lower than the required threshold *p*, if the vehicle continues adopting $X_c^{\mathbf{S}}$. Thus, no alternative routing plan in $\mathbf{XP^S}$ does not eliminate the risk of adopting a different routing plan with lower reward. Since $\mathbf{XP^S}$ only includes high-quality solutions, such very low-reward solution is not included in $\mathbf{XP^S}$ in current status **S** but may be adopted by the vehicle in the future, if $X_c^{\mathbf{S}}$ is still adopted as the current routing plan. Thus, $X^{\mathbf{S}}$ should be accepted as the current routing plan in order to avoid the risk of significant reward reduction.

If such solution $XP_{ic}^{\mathbf{S}}$ exists, we should estimate the probability $(pd(X_c^{\mathbf{S}}))$ that $X_c^{\mathbf{S}}$ is qualified to still be adopted as the current routing plan in the future. In other words, $pd(X_c^{\mathbf{S}})$ is the probability that $X_c^{\mathbf{S}}$'s in-time arrival probability can reach the value greater than or equal to the threshold *p* when the vehicle starts to depart to the first customer in $X_c^{\mathbf{S}}$ different from that in $XP_{ic}^{\mathbf{S}}$. We use Fig. 7 to demonstrate how this probability is defined.

The first two customers (3 and 4) after status **S** in $X_c^{\mathbf{S}}$ and $XP_{ic}^{\mathbf{S}}$ are identical. Thus, $pd(X_c^{\mathbf{S}})$ is the probability that the $X_c^{\mathbf{S}}$'s in-time arrival probability increases from "94%" in the current status to "$\geq$95%" when the vehicle finishes serving Customer 4 in Fig. 7(a). Similar to the second situation, $pd(X_c^{\mathbf{S}})$ can be formulated as:

$$pd(X_c^{\mathbf{S}}) = P(P(tb + t_{4,5} + t_{5,8} + st_5 \leqslant T_e^{\mathbf{S}} \mid tb = t_{2,3} + t_{3,4} + st_3 + st_4) \geqslant p).$$

The calculation of $pd(X_c^{\mathbf{S}})$ is similar to that in Table 2. If $X_c^{\mathbf{S}}$ is adopted as the current routing plan, the expected decrease in reward is defined as

$$ED = \min_{XP_{ic}^{\mathbf{S}} \in \mathbf{XP^S}} (1 - pd(X_c^{\mathbf{S}})) \times (RW(X_c^{\mathbf{S}}) - RW(XP_{ic}^{\mathbf{S}})) \tag{9a}$$

Subject to

$$IAP^{\mathbf{S}}(XP_{ic}^{\mathbf{S}}) \geqslant p \tag{9b}$$

$$(R^{\mathbf{S}}(X_c^{\mathbf{S}}))_1 = (R^{\mathbf{S}}(XP_{ic}^{\mathbf{S}}))_1 \tag{9c}$$

$$pd(X_c^{\mathbf{S}}) = P(P(tb^k(X_c^{\mathbf{S}}) + ta^k(X_c^{\mathbf{S}}) \leqslant T_e^{\mathbf{S}} \mid tb^k(X_c^{\mathbf{S}})) \geqslant p) \tag{9d}$$

Formula (9a) aims to obtain the minimum expected potential decreased reward (*ED*) if $X_c^{\mathbf{S}}$ is still adopted as the current routing plan. Formula (9b) ensures that the solutions in $\mathbf{XP^S}$ with the in-time arrival probability smaller than *p* have no opportunity to calculate *ED*. Formula (9c) ensures that the next to-be-visited customer in the solution $XP_{ic}^{\mathbf{S}}$ should be identical with that in solution $X_c^{\mathbf{S}}$, otherwise $XP_{ic}^{\mathbf{S}}$ has no opportunity to calculate *ED*. Formula (9d) is to calculate the probability $pd(X_c^{\mathbf{S}})$. If $ED > (RW(X_c^{\mathbf{S}}) - RW(X^{\mathbf{S}}))$, we replace $X_c^{\mathbf{S}}$ with $X^{\mathbf{S}}$ as the current routing plan. Otherwise, $X_c^{\mathbf{S}}$ is kept as the current routing plan. For the example in Fig. 7, assuming that $pd(X_c^{\mathbf{S}}) = 0.4$, $ED = 0.6 \times (140\text{–}115) = 15 < RW(X_c^{\mathbf{S}}) - RW(X^{\mathbf{S}}) = 140\text{–}120 = 20$. Thus $X_c^{\mathbf{S}}$ is kept as the current routing plan.

We use Formula (10) to express Algorithm 5 and the pseudocode of this algorithm is given in Appendix A.

$$(X_c^{\mathbf{S}}, \mathbf{S}, T_b^{\mathbf{S}}) = \textbf{Algorithm } 5\ (\mathbf{XP^S}, \mathbf{S}, X^{\mathbf{S}}, X_c^{\mathbf{S}}, T_e^{\mathbf{S}}) \tag{10}$$

The inputs include:

$\mathbf{XP^S}$: the preliminary screened solution pool obtained by Algorithm 4;
S: the current status (before updated);
$X^{\mathbf{S}}$: the selected solution obtained by Algorithm 4;
$X_c^{\mathbf{S}}$: the current adopted routing plan;
$T_e^{\mathbf{S}}$: the remaining time budget at the end of the status **S**.

The outputs include:

$X_c^{\mathbf{S}}$: updated current routing plan that the vehicle adopts to execute tasks;
S: updated status;

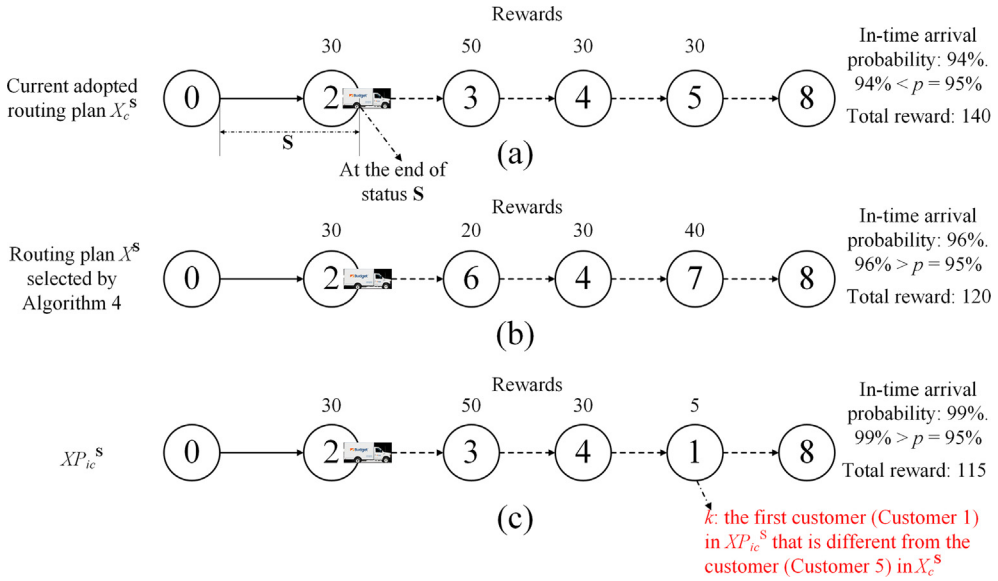


Fig. 7. An example of situation 3.

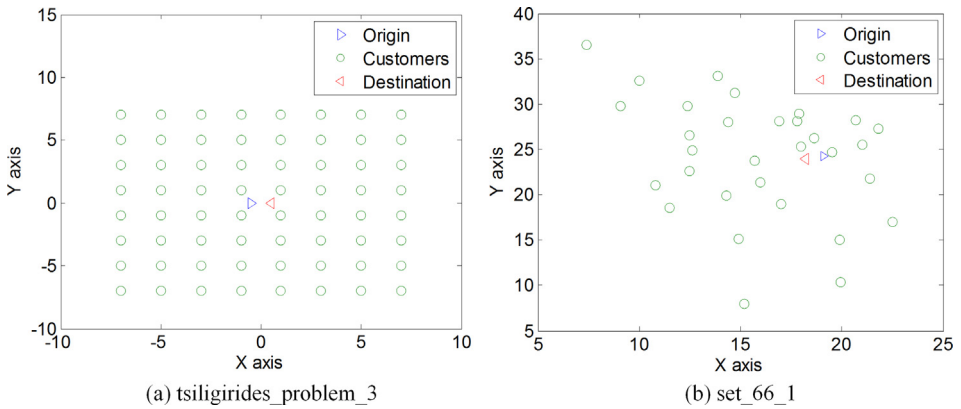


Fig. 8. The network topology of the benchmarks.

Table 3
Comparison results of the three algorithms.

Examples	Re-optimization versus strategic-level algorithm			SMPA versus strategic-level algorithm			SMPA versus re-optimization		
	Ns	Nif	Nid	Ns	Nif	Nid	Ns	Nif	Nid
T_33_50	42	25	33	50	3	47	45	14	41
T_33_80	49	22	29	47	4	49	37	25	38
A_50_400	66	28	6	74	9	17	51	25	24
A_50_600	53	37	10	83	5	12	60	17	23
S_66_60	66	30	4	89	9	2	62	22	16
S_66_90	62	29	9	83	4	13	62	18	20
A_100_600	68	25	7	86	13	1	56	21	23
A_100_800	65	27	8	91	7	2	63	25	12

Notes: Ns: number of trials that the routing plans obtained by the first algorithm are “superior” to those obtained by the second algorithm; Nif: number of trials that the routing plans obtained by the first algorithm are “inferior” to those obtained by the second algorithm; Nid: number of trials that the routing plans obtained by the first algorithm are identical with those obtained by the second algorithm.

Table 4
Late arrival rates of the routing plans obtained by the three algorithms (out of 100 trials).

Numerical examples	Late arrival rate		
	Strategic-level algorithm	Re-optimization	SMPA
T_33_50	1/100	0/100	0/100
T_33_80	2/100	1/100	2/100
A_50_400	2/100	1/100	0/100
A_50_600	6/100	1/100	2/100
S_66_60	3/100	4/100	0/100
S_66_90	2/100	1/100	0/100
A_100_600	4/100	0/100	0/100
A_100_800	4/100	2/100	3/100

Table 5
Computing times of the three algorithms in different periods.

Numerical examples	Algorithms	Computing time spent in preparing period (s)	Computing time spent in traveling and serving periods (s)	Computing time spent in decision making periods (s)
T_33_50	SLA	23.24	0	0
	ROA	18.73	0	Max: 16.50, Avg: 6.93
	SMPA	13.58	Max: 3.84, Avg: 1.01	Max: 0.38, Avg: 0.09
T_33_80	SLA	24.98	0	0
	ROA	19.36	0	Max: 24.00, Avg: 7.41
	SMPA	13.77	Max: 3.74, Avg: 1.16	Max: 0.25, Avg: 0.07
A_50_400	SLA	68.29	0	0
	ROA	51.38	0	Max: 50.35, Avg: 21.50
	SMPA	41.22	Max: 11.34, Avg: 4.51	Max: 1.72, Avg: 0.52
A_50_600	SLA	73.83	0	0
	ROA	56.28	0	Max: 54.74, Avg: 23.02
	SMPA	46.54	Max: 13.26, Avg: 4.71	Max: 2.01, Avg: 0.55
S_66_60	SLA	104.02	0	0
	ROA	78.70	0	Max: 74.96, Avg: 30.33
	SMPA	59.86	Max: 18.83, Avg: 5.58	Max: 2.69, Avg: 0.60
S_66_90	SLA	120.87	0	0
	ROA	84.19	0	Max: 79.23, Avg: 33.43
	SMPA	63.72	Max: 21.69, Avg: 6.11	Max: 2.72, Avg: 0.74
A_100_600	SLA	173.96	0	0
	ROA	126.38	0	Max: 118.63, Avg: 34.25
	SMPA	104.69	Max: 23.96, Avg: 6.80	Max: 6.11, Avg: 0.97
A_100_800	SLA	197.82	0	0
	ROA	138.00	0	Max: 134.86, Avg: 39.55
	SMPA	117.09	Max: 27.02, Avg: 7.39	Max: 6.30, Avg: 1.02

Notes: SLA: strategic-level algorithm; RO: re-optimization algorithm; SMPA: simulation-aided multiple plan approach (proposed in this paper).

$T_b^{\mathbf{S}}$: the remaining time budget when the updated status **S** starts.

5. Numerical examples

5.1. Data setting

Chao (1996b) and Tsiligirides (1984) proposed standard benchmarks of the orienteering problem to test the performances of different algorithms. We select four benchmarks (tsiligirides_problem_3_budget_050, tsiligirides_problem_3_budget_080, set_66_1_060, set_66_1_090) from the website (http://www.mech.kuleuven.be/en/cib/op#section-2). The time budgets in the four benchmarks are "50", "80", "60", and "90", respectively. Fig. 8 shows the departure locations, destinations, and customers that may be visited by the vehicles in the benchmarks of tsiligirides_problem_3 and set_66_1. The network topology of tsiligirides_problem_3_budget_050 and tsiligirides_problem_3_budget_080 is presented in Fig. 8(a), and that of set_66_1_060 and set_66_1_090 is shown in Fig. 8(b).However, these benchmarks cannot be directly applied to test the performance of the proposed algorithm in this paper, because the travel time in the benchmarks are all deterministic. Thus, we modify the travel times as stochastic variables. The travel times are set to be log-normal distributions ($t_{i,j} \sim \text{lognormal}(\mu_{i,j}, \sigma_{i,j}^2)$), which is a reasonable assumption based on Guessous et al. (2014). Let $\mu_{i,j} = \log(\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2})$, where x_i and y_i are the horizontal and vertical coordinates of all customer locations, and $\sigma_{i,j}$ ($i, j \in \mathbf{N}$)

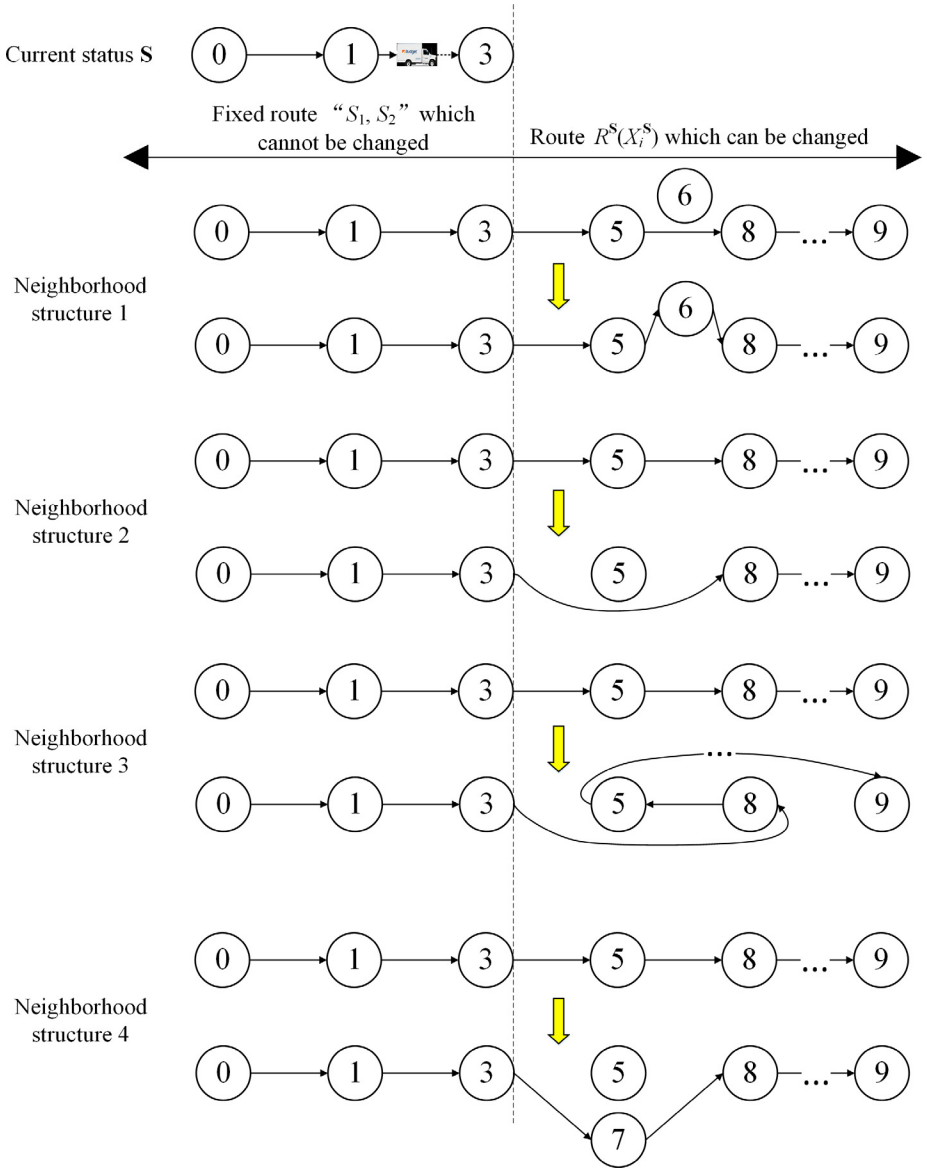


Fig. 9. Four neighborhood structures.

are all drawn from normal distributions with the mean of "0.15" and the standard deviation of "0.005". The practice usually requires that the minimum in-time arrival probability (p) should be close to "100%". However, if we set p very close to 100%, the computer needs to run a large number of trials to show the "late-arrival phenomena". Thus, p should not be too close to "100%". We prescribe that the required minimum in-time arrival probability is 95% in all numerical examples (p = 95%), which is a reasonable value. We re-name the four numerical examples as T_33_50, T_33_80, S_66_60, and S_66_90. The first number in the notation is the number of customers and the second number represents the time budget.The modified benchmarks do not have stochastic service times and the scales of the benchmarks are not large enough. Thus, besides the modified standard benchmarks, we proposed four additional numerical examples which are randomly generated by the computer. The travel times are set to follow log-normal distributions ($t_{i,j} \sim \text{lognormal}(\mu_{i,j},\sigma_{i,j}^2)$) and the service times are set to follow normal distributions $st_i \sim \text{normal}(\mu_i,\sigma_i^2)$. The horizontal coordinates (x_i, $i \in \mathbf{CT}$) and vertical coordinates (y_i, $i \in \mathbf{CT}$) of all customer locations are randomly generated from normal distributions with the mean of "0" and the standard deviation of "20". Let $\mu_{i,j} = \log(\sqrt{(x_i-x_j)^2 + (y_i-y_j)^2})$. $\sigma_{i,j}$ (i, $j \in \mathbf{N}$) are all drawn from normal distributions with the mean of "0.15" and the standard deviation of "0.005". μ_i($i \in \mathbf{CT}$) are drawn from the uniform distribution within the interval [5, 10] and σ_i($i \in \mathbf{CT}$) are generated from normal distributions with the mean of "1" and the standard deviation of "0.01".

In these four numerical examples, the departure location and the destination are the same place. The four numerical examples are denoted by "A50_400", "A50_600", "A100_500" and "A100_800". In "A50_400", "50" is the number of customers and "400" is the time budget that allows the vehicle to execute the transportation tasks. The required minimum in-time arrival probability is 95%

(p = 95%) in all numerical examples.

The designed 8 numerical examples have different scales (number of customers), different time budgets, and different travel and service times, so that the proposed algorithm is tested in various scenarios. In order to verify the effectiveness of the proposed SMPA, we compare the SMPA with a strategic-level algorithm and another operational-level algorithm, re-optimization, which is commonly used to solve the dynamic vehicle routing problem in the literature (Pillac et al., 2013). The strategic-level algorithm determines an *a priori* vehicle routing plan for the stochastic orienteering problem before the vehicle departs from the departure location and does not adjust the routing plan when the vehicle is executing the transportation and serving tasks. We use the Tabu Search algorithm (Osman, 1993) to implement the strategic-level algorithm, whose effectiveness is verified by Gendreau et al. (1998b). The details of the Tabu Search is give in the Appendix A (Algorithm 6). For the re-optimization algorithm, the Tabu Search algorithm (Algorithm 6) is adopted to re-optimize the vehicle routing for the remaining customers every time the vehicle finishes serving a customer. The re-optimization algorithm always adopts the optimal solution found by the algorithm as the current routing plan at the end of each status. The re-optimization algorithm has the "myopia" problem that is introduced in Section 4. Moreover, the re-optimization algorithm never utilizes the leisure time when the vehicle is on the way or serving a customer and all time is spent in the prompt decision making periods when the vehicle finishes serving a customer. Both the strategic-level algorithm and the re-optimization algorithm are achieved by Tabu Search (TS) algorithm due to comparison fairness, because the TS is also implemented for SMPA.

5.2. Running conditions

The algorithms are programmed in Matlab R2014a on a Dell computer with processor Intel(R) Core(TM) i7-4790 CPU @ 3.60 GHz and 8 GB RAM.

5.3. Experimental results

For each numerical example, we run the three algorithms, namely strategic-level algorithm, re-optimization, and SMPA for 100 trials. In each trial, the computer generates a group of travel times and service times as real values if the vehicle traverses the links and serves the customers. The vehicle executes the transportation and serving task based on the routing plans generated by each of the three algorithms, namely SMPA, re-optimization, and the strategic-level algorithm. We conduct three comparisons in each trial, which are re-optimization versus strategic-level algorithm, SMPA versus strategic-level algorithm, and SMPA versus re-optimization. We record the number of trials that the routing plans obtained by the first algorithm that are "superior to", "inferior to", and "identical with" those obtained by the second algorithm. We define a routing plan as "superior to" another routing plan if at least one of the two conditions is satisfied: 1) the vehicle arrives at the planned destination within the time budget if the vehicle adopts the first routing plan while the vehicle cannot reach the planned destination if the vehicle adopts the second routing plan, 2) the vehicle can reach the planned destination if it adopts both of the first and second routing plan, but the reward of first routing plan is greater than that of the second routing plan.

Table 3 presents the comparison results of the three algorithms, showing the number of trials that the first algorithm obtains superior, inferior and identical routing plans compared against the second algorithm. Take "A_100_600" as an example, re-optimization algorithm obtains 68, 25 and 7 routing plans that are respectively superior to, inferior to, and identical with that obtained by the strategic-level algorithm, while the SMPA obtains 86, 13 and 1 routing plans that are respectively superior to, inferior to, and identical with that obtained by the strategic-level algorithm.

In Table 3, from the comparisons of **re-optimization versus strategic-level algorithm** and **SMPA versus strategic-level algorithm**, we find that both re-optimization and SMPA outperform the strategic-level algorithm because the number of "superior" routing plans are significantly larger than the number of "inferiors" obtained by the two operational-level algorithms for all numerical examples. However, the differences between the number of "superior" routing plans and the number of "inferiors" in the comparison of SMPA versus strategic-level algorithm is much larger than those in the comparison of re-optimization versus strategic-level algorithm. This indicates that the SMPA outperforms the strategic-level algorithm more significantly than the re-optimization algorithm does in terms of the solution quality. Moreover, in general, as the problem scale increases, the difference between the number of "superiors" and the number of "inferiors" in the comparison of SMPA versus strategic-level algorithm becomes larger. This implies that the SMPA may outperform the strategic-level algorithm more significantly in solving larger-scale problems. The direct comparison of **SMPA versus re-optimization** also indicates that the SMPA outperforms the re-optimization in terms of the overall solution quality, implied from the phenomenon that the number of "superiors" is significantly larger than the number of "inferiors" for all numerical examples. This is because the novel "myopia prevention" strategy embedded in SMPA can increase the expected reward collected by the vehicle.

Table 4 shows the numbers of late arrivals out of the 100 trials from the three algorithms, the strategic-level algorithm, the re-optimization algorithm, and the SMPA. That is the number of trials in which the vehicle cannot reach the planned destination within the time budget when the vehicle adopts the routing plans obtained by the three algorithms. Take "S_66_60" as an example, the vehicle cannot reach the planned destination within the time budget for 3, 4 and 0 trials out of the total 100 trials if the vehicle adopts the routing plans obtained by the strategic-level algorithm, the re-optimization algorithm and the SMPA, respectively.

We can imply from Table 4 that, in general, both two operational-level algorithms can reduce the late-arrival probability. The late arrival rates of the re-optimization algorithm for all numerical examples except "S_66_60" are no larger than those of the strategic-level algorithm, and the late arrival rates of the SMPA are all no larger than those of the strategic-level algorithm. The late arrival rates of both the SMPA and the re-optimization algorithms are smaller than those of the strategic-level algorithm for seven out of

eight numerical examples.

Table 5 presents the computing time of the three algorithms spent in the three periods, preparing period, traveling and serving periods, and decision making periods. In the preparing period, the strategic-level algorithm spends the longest computing time among the three algorithms for all tested numerical examples. Since the strategic-level algorithm does not spend any time in the traveling and serving periods and the subsequent decision making periods, it has to fully utilize the computing time in the preparing period to obtain a high-quality solution and thus spends the longest time in that period. The two operational-level algorithms, re-optimization and SMPA, use less computing time in the preparing period because they will adjust the routing plan in real time and do not necessarily need to obtain a very high-quality *a priori* routing plan in this period. In the traveling and serving periods, the strategic-level algorithm and re-optimization algorithm do not spend any computing time. Only the SMPA utilizes computing time in this period.

Because there are multiple traveling and serving periods in a routing plan, we record the maximum computing time spent in all traveling and serving periods as well as calculate the average computing time spent in all traveling and serving periods. The computing time spent by the SMPA in this period is reasonable. The maximum time spent in all traveling and serving periods for the largest-scale numerical example, A_100_800, is 27.02 s, less than 30 s. Note that the computing time in this period can be changed by adjusting the number of iterations or the number of solutions generated. The computing time spent in this period should not exceed the time length of the current traveling and serving period. During the decision making periods, the operational-level algorithms have to determine the next to-be-visited customer by determining the routing plan at the end of each status. The computing time spent by the re-optimization algorithm is significantly longer than that spent by the SMPA, especially in solving large-scale problems (e.g. A_100_800). This is because the re-optimization algorithm needs to re-optimize the routing plan each time the vehicle finishes serving a customer while the SMPA only needs to select a solution from a pre-generated solution pool to determine the routing plan after the vehicle finishes serving a customer. From Table 5, the maximum computing time of the SMPA in all decision making periods for the largest-scale problem (A_100_800) is only 6.30 s, which is prompt enough for most scenarios.

In this table, we have to explain one abnormal phenomenon. The maximum computing time of the SMPA in all decision making periods for T_33_50 is 3.84 s, which is longer than that spent by the example T_33_80 (3.74 s). This case seems that an instance with the same number of nodes but with lower time budget spends longer time. This is possible. Algorithms 4 and 5 are used in the decision making periods. When the next to-be-visited customers in solutions $X^{\mathbf{S}}$ and $X_c^{\mathbf{S}}$ are different, Algorithm 5 spends much more time than Algorithms 4 because the most computing time is to calculate the probability "*pd*" (see the algorithm in Table 2). The number of calculations of "*pd*" is identical with the number of solutions, whose next to-be-visited customer is identical with that in solution $X_c^{\mathbf{S}}$ (see Formulas 8c and 9c), in $\mathbf{XP}^{\mathbf{S}}$. This number of solutions has randomness and does not entirely depend on the scale of the problem. Thus, it is possible that the computing time of T_33_50 in some decision making periods is larger than that of T_33_80.

5.4. Summary of contributions

This section summarizes the good performance of the proposed simulation-aided multiple plan approach (SMPA) in solving operational stochastic orienteering problem (OSOP). Based on the experimental results, we make the following contribution statements.

1) The SMPA outperforms the strategic-level algorithm more significantly than another commonly used operational-level algorithm, called the re-optimization algorithm, in solving the OSOP. This is because a novel myopia prevention strategy is embedded in SMPA so that the expected reward of the obtained routing plan can be increased.
2) The SMPA can reduce the late-arrival risk compared with the strategic-level algorithm. The strategic-level algorithm determines an *a priori* routing plan in the planning stage, while the SMPA can adjust the routing plan in real time. After serving certain customers, when the vehicle has a low in-time arrival probability if continuing to adopt the *a priori* routing plan, the vehicle can adjust the routing plan to increase the in-time arrival probability. Thus, in general, the routing plans obtained by SMPA have higher in-time arrival probabilities than those obtained by the strategic-level algorithm.
3) The SMPA can save computing time spent in the prompt decision making periods when compared with the re-optimization algorithm. The SMPA fully utilizes the computer's leisure time in the traveling and serving periods to obtain a solution pool to prepare the computing activity in the prompt decision making periods. In contrast, the re-optimization algorithm does not utilize this leisure time and thus spends a longer time for re-optimizing the routing plan in the decision making periods.

6. Conclusions

This paper studies the operational-level stochastic orienteering problem, in which the vehicle can adjust the routing plan in real time. A real-time adjustment strategy called Simulation-Aided Multiple Plan Approach (SMPA) is proposed to direct the vehicle to execute the transporting and serving tasks. The SMPA uses Monte Carlo simulation to evaluate the in-time arrival probability and uses a multiple plan approach to determine the real-time routing plan. A novel myopia prevention strategy is embedded into the multiple plan approach to improve the solution quality. We use eight numerical examples to compare the proposed algorithm with the strategic level-algorithm and the re-optimization algorithm. The numerical experimental results show that the proposed real-time adjustment strategy (SMPA) obtains higher-quality routing plans than the strategic-level algorithm and the re-optimization algorithm. The SMPA outperforms the strategic-level algorithm more significantly in solving larger-scale problems. Additionally, based on the comparison results of the late-arrival probability, we find that the SMPA can reduce the late-arrival probability compared with the

strategic-level algorithm. Finally, the SMPA can efficiently utilizes the time when the vehicle is traveling or serving a customer to generate routing plans preparing for the prompt determination of the future routing plan when the vehicle finishes serving a customer. The computing time of the SMPA in different periods are all reasonable in the studied scenarios. This research indicates the promising application of SMPA to real-time vehicle routing adjustment in a wide array of transportation and logistics research problems.

7. Limitations and future work

The proposed real-time adjustment strategy successfully solves the operational-level stochastic orienteering problem. However, this paper has limitations, which will be addressed in our future work.

- This paper focuses on the one-vehicle stochastic orienteering problem. In our future work, we will propose real-time adjustment strategies for the operational-level team orienteering problem in which multiple vehicles are dispatched to execute the transporting and serving tasks collaboratively.
- Another extended interesting problem we will study is the operational-level dynamic and stochastic orienteering problem. In this problem, the customer requests occur in random locations and random time when the vehicle is executing the tasks, and the travel time and service time and/or each customer's reward are stochastic. We will adapt the SMPA to handle such highly uncertain orienteering problem.
- Finally, we will apply the proposed algorithm in practical problems, such as the fuel delivery problem, the single-ring design problem when building telecommunication networks, the tourist trip design problem, and unmanned aircraft and submarine surveillance problem.

Appendix A – Algorithms used in this paper

Algorithm 1 Monte Carlo simulation to estimate the in-time arrival probability$IAP^{\mathbf{S}}(X)$ = **Algorithm 1** (X, $T_e^{\mathbf{S}}$, **t**, **st**, NSR)

Initialize $ni = 0$ (the index of current simulation replication), $nn = 0$ (number of in-time arrivals);
Do while $ni < NSR$
  Randomly generate the travel times ($t_{i,j}^{ni}$) for all links $(i, j) \in L(U^{\mathbf{S}}(X))$ and service times (st_i^{ni}) for all customers $i \in C(R^{\mathbf{S}}(X))$ based on their distributions;
  Calculate the total task execution time$tt^{ni} = \sum_{(i,j)\in L(U^{\mathbf{S}}(X))} t_{i,j}^{ni} + \sum_{i\in C(R^{\mathbf{S}}(X))} st_i^{ni}$;
  If $tt^{ni} \le T_e^{\mathbf{S}}$
    $nn = nn + 1$;
  End if
  $ni = ni + 1$;
End do
Calculate the in-time arrival probability $IAP^{\mathbf{S}}(X) = nn/NSR$.

Algorithm 2 Tabu Search to generate the solution pool $\mathbf{X}^{\mathbf{S}}$ in status **S**
$\mathbf{X}^{\mathbf{S}}$ = **Algorithm 2** ($T_b^{\mathbf{S}}$, **t**, **st**)

Initialize the total number of periods (NP), number of iterations in each period (NIP), number of candidate solutions (CN), number of solutions (NS) assigned into the solution pool for each iteration;
Set $icp = 0$ (the index of the current period), $ici = 0$ (the index of current iteration in each period), $X_{current} = X_0^{\mathbf{S}}$ as an initial routing plan; % $X_0^{\mathbf{S}}$ is randomly generated by the computer. First, a group of to-be-visited customers are randomly selected from the all customers **CT**. Then the selected customers are also randomly permuted to form the routing sequence $X_0^{\mathbf{S}}$.
Set the tabu list as empty: $TL = \varnothing$;
Do while $icp < NP$
  $icp = icp + 1$;
  Randomly generate a group of travel times $\mathbf{t}^{icp} = \{t_{i,j}^{icp} \mid i,j \in \mathbf{N}\}$ and service times $\mathbf{st}^{icp} = \{st_i^{icp} \mid i \in \mathbf{CT}\}$ based on their distributions;
  Do while $ici < NIP$
    Generate CN candidate solutions $\mathbf{X} = \{X_1, X_2, \ldots, X_{CN}\}$ of $X_{current}$'s neighbors (see Appendix B);
    Calculate $\{Z(X_1), Z(X_2), \ldots, Z(X_{CN})\}$ based on Formula (3) and record the subscript opt, where $Z(X_{opt}) = \max\{Z(X_1), Z(X_2), \ldots, Z(X_{CN})\}$;

Algorithm 2 Tabu Search to generate the solution pool $\mathbf{X^S}$ in status **S**
$\mathbf{X^S}$ = **Algorithm 2** ($T_b^{\mathbf{S}}$, **t**, **st**)

Put *NS* solutions with maximum *Z* values into the solution pool $\mathbf{X^S}$;
Do while X_{opt} is in tabu list
$\mathbf{X} = \mathbf{X} \setminus X_{opt}$;
X_{opt} = argmax $\{Z(X_1), Z(X_2),\ldots, Z(X_{CN})\}$;
End do
$X_{current} = X_{opt}$;
Put X_{opt} into the tabu list *TL*;
ici = *ici* + 1;
End do
End do

Algorithm 3 Compatible solution transfer algorithm$\mathbf{X^S}$ = **Algorithm 3** ($\mathbf{X^S}$, $\mathbf{X^{S^l}}$)

For i = 1: $|\mathbf{X^{S^l}}|$ % The number of routing plans in $\mathbf{X^{S^l}}$
If $S_2 = (R^{\mathbf{S}^l}(X_i^{\mathbf{S}^l}))_1$
$\mathbf{X^S} = \mathbf{X^S} \cup$
$\{X_i^{\mathbf{S}^l}\}$; % put $X_i^{\mathbf{S}^l}$ into $\mathbf{X^S}$
End if
End for

Algorithm 4 Solution selection algorithm
($X^{\mathbf{S}}$, $\mathbf{XP^S}$, $T_e^{\mathbf{S}}$) = **Algorithm 4** ($\mathbf{X^S}$, $T_b^{\mathbf{S}}$, $t^{\mathbf{S}}$, *p*, **t**, **st**)

$T_e^{\mathbf{S}} = T_b^{\mathbf{S}} - t^{\mathbf{S}}$;
Set $\mathbf{X} = \mathbf{X^S}$;
For j = 1: $|\mathbf{X^S}|$
% Preliminary screen: remove the solutions whose in-time arrival probabilities are obviously less than *p*.
$IAP^{\mathbf{S}}(X_j^{\mathbf{S}})$ = **Algorithm** 1 ($X_j^{\mathbf{S}}$, $T_e^{\mathbf{S}}$, **t**, **st**, *NPS*); %$X_j^{\mathbf{S}}$ is the *j*th solution in $\mathbf{X^S}$
If $IAP^{\mathbf{S}}(X_j^{\mathbf{S}}) < p'$ % p' is much smaller than *p*
$\mathbf{X} = \mathbf{X} \setminus X_j^{\mathbf{S}}$;
End If
End for
Sequence all solutions in **X** in order of descending rewards and we get $\mathbf{X} = (X_{(1)}, X_{(2)}, \ldots, X_{(n)})$ with rewards $\mathbf{RW(X)} = (RW_{(1)}, RW_{(2)}, \ldots, RW_{(n)})$;
% *n*: number of solutions in **X**
i = 1;
% *i*: the index of a solution to be measured by Monte Carlo simulation
Do
$IAP^{\mathbf{S}}(X_{(i)})$ = **Algorithm** 1 ($X_{(i)}$, $T_e^{\mathbf{S}}$, **t**, **st**, *NMP*);
$i = i+1$;
While $IAP^{\mathbf{S}}(X_{(i)}) < p$
$X^{\mathbf{S}} = X_{(i)}$;
$\mathbf{XP^S} = \mathbf{X}$;

Note: we set the number of preliminary simulation replications as 100 ($NPS = 100$). Then we calculate the sample standard deviation ($std(t_{total})$) of the 100 simulation replications of total task execution time (t_{total}). The number of simulation replications for estimation of in-time arrival probability is set to $NMP = 1500\max(std(t_{total}) - 5, 1)$ because this number of simulation replications is sufficient to ensure the accuracy of in-time arrival probability estimation (most are within the interval $[-5\%\ IAP^{\mathbf{S}}(X),\ +5\%\ IAP^{\mathbf{S}}(X)]$).

Algorithm 5 Algorithm for updating the current routing plan adopted by the vehicle
$(X_c^{\mathbf{S}}, \mathbf{S}, T_b^{\mathbf{S}}) =$ **Algorithm 5** $(\mathbf{XP}^{\mathbf{S}}, \mathbf{S}, X^{\mathbf{S}}, X_c^{\mathbf{S}}, T_e^{\mathbf{S}})$

$a = (R^{\mathbf{S}}(X_c^{\mathbf{S}}))_1$, $b = (R^{\mathbf{S}}(X^{\mathbf{S}}))_1$;
$\mathbf{S}^a$ = ("S_1, S_2", a, $S_3 \backslash a$); % it is the new status if $X_c^{\mathbf{S}}$ is still kept as the current routing plan.
$\mathbf{S}^b$ = ("S_1, S_2", b, $S_3 \backslash b$); % it is the new status if $X_c^{\mathbf{S}}$ is replaced by $X^{\mathbf{S}}$.
If $a = b$ % the first situation
  $\mathbf{S} = \mathbf{S}^a$;
Else
  If $RW(X_c^{\mathbf{S}}) \le RW(X^{\mathbf{S}})$ % the second situation
    Use Monte Carlo simulation to estimate the expected potential increased reward (EI) based on Formula (8);
    If $EI < RW(X^{\mathbf{S}}) - RW(X_c^{\mathbf{S}})$
     $\mathbf{S} = \mathbf{S}^b$;
     $X_c^{\mathbf{S}} = X^{\mathbf{S}}$;
    Else
     $\mathbf{S} = \mathbf{S}^a$;
    End if
Else % consider the third situation that $RW(X_c^{\mathbf{S}}) \ge RW(X^{\mathbf{S}})$ & $IAP^{\mathbf{S}}(X_c^{\mathbf{S}}) < p$
  Use Monte Carlo simulation to estimate the expected potential decreased reward (ED) based on Formula (9);
    If $ED > RW(X_c^{\mathbf{S}}) - RW(X^{\mathbf{S}})$
     $\mathbf{S} = \mathbf{S}^b$;
     $X_c^{\mathbf{S}} = X^{\mathbf{S}}$;
    Else
     $\mathbf{S} = \mathbf{S}^a$;
    End if
  End if
End if
$T_b^{\mathbf{S}} = T_e^{\mathbf{S}}$;

Algorithm 6 Strategic-level algorithm (Tabu Search)
$X_{best} =$ **Algorithm 6** $(T_b^{\mathbf{S}}, \mathbf{t}, \mathbf{st})$

Initialize the total number of iterations (NI), number of candidate solutions (CN). Set $ni = 0$ (the index of the current iteration), $X_{current} = X_0^{\mathbf{S}}$ as an initial routing plan.
$X_{best} = X_0^{\mathbf{S}}$ (record the best solution found), $RW_{best} = 0$ (record the reward of X_{best});Set the tabu list as empty: $TL = \varnothing$;
Do while $ni < NI$
  $ni = ni + 1$;
  Generate CN candidate solutions $\{X_1, X_2, \ldots, X_{CN}\}$ of $X_{current}$'s neighbors (see Appendix B).
  Use Algorithm 1 to simulate the in-time arrival probabilities of all CN solutions: $IAP(X_i) =$ **Algorithm 1** $(X_i, T, \mathbf{t}, \mathbf{st}, NSR)$, for all $i = 1, 2, \ldots, CN$;
  Let $\mathbf{M} = \{ X_j \mid$ all j that $IAP(X_j) \ge p\}$;
  Calculate $RW(X_j)$ for all $X_j \in \mathbf{M}$ and record the subscript opt, where $RW(X_{opt}) = \max RW(X_j)$, $X_j \in \mathbf{M}$;
  If $RW(X_{opt}) > RW_{best}$
    $X_{best} = X_{opt}$;
    $RW_{best} = RW(X_{opt})$;
  End if
  Do while X_{opt} is in tabu list
    $\mathbf{M} = \mathbf{M} \setminus X_{opt}$;
    $X_{opt} = \text{argmax}\ RW(X_j)$, $X_j \in \mathbf{M}$;
  End do
  $X_{current} = X_{opt}$;

Algorithm 6 Strategic-level algorithm (Tabu Search)
X_{best} = **Algorithm 6** ($T_b^{\mathbf{S}}$, **t**, **st**)

Put X_{opt} into the tabu list *TL*;
End do

Appendix B – The neighborhood structures used in Tabu Search

We designed four neighborhood structures for the iteration in Tabu Search. Assuming that the current status is **S**, $X_i^{\mathbf{S}}$ is a solution generated in the status **S**. We use $X_i^{\mathbf{S}}$ as an example to show the neighborhood structures (Fig. 9)

- Neighborhood 1: randomly select a customer location x_i, $x_i \in S_3$ which is not visited, and insert it to the route $U^{\mathbf{S}}(X_i^{\mathbf{S}})$. We insert this customer location to the position where the increase of task execution time is the minimum among all insertion positions in the route.
- Neighborhood 2: remove one customer location from the route $R^{\mathbf{S}}(X_i^{\mathbf{S}})$. The vehicle will not visit this customer location after removal.
- Neighborhood 3: reverse the sequence of nodes between two selected customer positions in the route $R^{\mathbf{S}}(X_i^{\mathbf{S}})$ while keeping destination unchanged (the famous 2-opt neighborhood).
- Neighborhood 4: simultaneously insert a customer location $x_i \in S_3$ into the route $U^{\mathbf{S}}(X_i^{\mathbf{S}})$ and delete another customer location from the route $R^{\mathbf{S}}(X_i^{\mathbf{S}})$. Note that the insertion position should also be the one where the increased travel time is the minimum among all insertion positions in the route.

Appendix C. Supplementary data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.tre.2018.05.004.

References

Archetti, Claudia, Hertz, Alain, Speranza, Maria Grazia, 2007. Metaheuristics for the team orienteering problem. J. Heuristics 13 (1), 49–76.
Archetti, Claudia, Feillet, Dominique, Hertz, Alain, Speranza, Maria Grazia, 2009. The capacitated team orienteering and profitable tour problems. J. Operat. Res. Soc. 60 (6), 831–842.
Baffo, Ilaria, Carotenuto, Pasquale, Storchi, Giovanni, 2015. A genetic algorithm to design touristic routes in a bike sharing system. In: Proc. of 14th International Conference on Modelling and Applied Simulation, MAS 2015, pp 196–205. ISBN 978-88-97999-59-1.
Bent, Russell W., Van Hentenryck, Pascal, 2004. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. Oper. Res. 52 (6), 977–987.
Bérubé, Je-François, Gendreau, Michel, Potvin, Je-Yves, 2009. An exact ε-constraint method for bi-objective combinatorial optimization problems: application to the traveling salesman problem with profits. Eur. J. Oper. Res. 194 (1), 39–50.
Butt, Steven E., Cavalier, Tom M., 1994. A heuristic for the multiple tour maximum collection problem. Comput. Oper. Res. 21 (1), 101–111.
Butt, Steven E., Ryan, David M., 1999. An optimal solution procedure for the multiple tour maximum collection problem using column generation. Comp. Oper. Res. 26 (4), 427–441.
Campbell, Ann M., Gendreau, Michel, Thomas, Barrett W., 2011. The orienteering problem with stochastic travel and service times. Ann. Oper. Res. 186 (1), 61–81.
Campos, Vicente, Martí, Rafael, Sánchez-Oro, Jesús, Duarte, Abraham, 2014. GRASP with path relinking for the orienteering problem. J. Operat. Res. Soc. 65 (12), 1800–1813.
Chao, I.-Ming, Golden, Bruce L., Wasil, Edward A., 1996a. The team orienteering problem. Eur. J. Oper. Res. 88 (3), 464–474.
Chao, I.-Ming, Golden, Bruce L., Wasil, Edward A., 1996b. A fast and effective heuristic for the orienteering problem. Eur. J. Oper. Res. 88 (3), 475–489.
Chekuri, Chandra, Kumar, Amit, 2004. Maximum coverage problem with group budget constraints and applications. In: Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. Springer, Berlin Heidelberg, pp. 72–83.
Chen, Cen, Cheng, Shih-Fen, Gunawan, Aldy, Misra, Archan, Dasgupta, Koustuv, Chander, Deepthi, 2014. Traccs: a framework for trajectory-aware coordinated urban crowd-sourcing. In: Second AAAI Conference on Human Computation and Crowdsourcing.
De Falco, Ivanoe, Scafuri, Umberto, Tarantino, Ernesto, 2015. A multiobjective evolutionary algorithm for personalized tours in street networks. In: European Conference on the Applications of Evolutionary Computation. Springer, Cham.
Evers, Lanah, Barros, Ana Isabel, Monsuur, Herman, Wagelmans, Albert, 2014a. Online stochastic UAV mission planning with time windows and time-sensitive targets. Eur. J. Oper. Res. 238 (1), 348–362.
Evers, Lanah, Glorie, Kristiaan, Ster, Suzanne Van Der, Barros, Ana Isabel, Monsuur, Herman, 2014b. A two-stage approach to the orienteering problem with stochastic weights. Comput. Oper. Res. 43, 248–260.
Feillet, Dominique, Dejax, Pierre, Gendreau, Michel, 2005. Traveling salesman problems with profits. Transport. Sci. 39 (2), 188–205.
Fischetti, Matteo, Gonzalez, Juan Jose Salazar, Toth, Paolo, 1998. Solving the orienteering problem through branch-and-cut. INFORMS J. Comput. 10 (2), 133–148.
Fomin, Fedor V., Lingas, Andrzej, 2002. Approximation algorithms for time-dependent orienteering. Inf. Process. Lett. 83 (2), 57–62.
Garcia, Ander, Olatz, Arbelaitz, Pieter, Vansteenwegen, Wouter, Souffriau, Maria, Teresa Linaza, 2010. Hybrid approach for the public transportation time dependent orienteering problem with time windows. In: International Conference on Hybrid Artificial Intelligence Systems, pp. Springer–Berlin Heidelberg.
Gendreau, Michel, Laporte, Gilbert, Semet, Frederic, 1998a. A branch-and-cut algorithm for the undirected selective traveling salesman problem. Networks 32 (4), 263–273.
Gendreau, Michel, Laporte, Gilbert, Semet, Frédéric, 1998b. A tabu search heuristic for the undirected selective travelling salesman problem. Eur. J. Oper. Res. 106 (2), 539–545.
Golden, Bruce L., Levy, Larry, Vohra, Rakesh, 1987. The orienteering problem. Naval Res. Logist. 34 (3), 307–318.
Guessous, Younes, Aron, Maurice, Bhouri, Neila, Cohen, Simon., 2014. Estimating travel time distribution under different traffic conditions. Transp. Res. Procedia 3,

339–348.
Gunawan, Aldy, Lau, Hoong Chuin, Vansteenwegen, Pieter, Lu, Kun, 2017. Well-tuned algorithms for the team orienteering problem with time windows. J. Operat. Res. Soc. 68 (8), 861–876.
Gunawan, Aldy, Lau, Hoong Chuin, Vansteenwegen, Pieter, 2016. Orienteering problem: a survey of recent variants, solution approaches and applications. Eur. J. Oper. Res. 255 (2), 315–332.
Gupta, Anupam, Krishnaswamy, Ravishankar, Nagarajan, Viswanath, Ravi, R., 2012. Approximation algorithms for stochastic orienteering. In: Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms. SIAM.
Ilhan, Taylan, Iravani, Seyed M.R., Daskin, Mark S., 2008. The orienteering problem with stochastic profits. IIE Trans. 40 (4), 406–421.
Jozefowiez, Nicolas, Glover, Fred, Laguna, Manuel, 2008. Multi-objective meta-heuristics for the traveling salesman problem with profits. J. Mathem. Modell. Algorithms 7 (2), 177–195.
Kantor, Marisa G., Rosenwein, Moshe B., 1992. The orienteering problem with time windows. J. Operat. Res. Soc. 43 (6), 629–635.
Kataoka, Seiji, Morito, Susumu, 1988. An algorithm for single constraint maximum collection problem. J. Oper. Res. Soc. Jpn. 31 (4), 515–530.
Ke, Liangjun, Archetti, Claudia, Feng, Zuren, 2008. Ants can solve the team orienteering problem. Comput. Ind. Eng. 54 (3), 648–665.
Kobeaga, Gorka, Merino, María, Lozano, Jose A., 2017. An efficient evolutionary algorithm for the orienteering problem. Comput. Oper. Res.
Labadie, Nacima, Mansini, Renata, Melechovský, Jan, Calvo, Roberto Wolfler, 2012. The team orienteering problem with time windows: an lp-based granular variable neighborhood search. Eur. J. Oper. Res. 220 (1), 15–27.
Laporte, Gilbert, Martello, Silvano, 1990. The selective travelling salesman problem. Discrete Appl. Math. 26 (2–3), 193–207.
Lau, Hoong Chuin, Yeoh, William, Varakantham, Pradeep, Nguyen, Duc Thien, Chen, Huaxing, 2012. Dynamic stochastic orienteering problems for risk-aware applications. arXiv preprint arXiv:1210.4874.
Leifer, Adrienne C., Rosenwein, Moshe B., 1994. Strong linear programming relaxations for the orienteering problem. Eur. J. Oper. Res. 73 (3), 517–523.
Li, Jin, Wu, Qinmin, Li, Xueqian, Zhu, Daoli, 2010. Study on the time-dependent orienteering problem. In: 2010 International Conference on E-Product E-Service and E-Entertainment (ICEEE). IEEE.
Liang, Yun-Chia, Smith, Alice E., 2006. An ant colony approach to the orienteering problem. J. Chin. Inst. Ind. Eng. 23 (5), 403–414.
Liao, Chen-Chih, Hsu, Cheng-Hsin, 2013. A detour planning algorithm in crowdsourcing systems for multimedia content gathering. In: Proceedings of the 5th Workshop on Mobile Video. ACM.
Osman, Ibrahim Hassan, 1993. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. Ann. Oper. Res. 41 (4), 421–451.
Ostrowski, Krzysztof, Karbowska-Chilinska, Joanna, Koszelew, Jolanta, Zabielski, Pawel, 2017. Evolution-inspired local improvement algorithm solving orienteering problem. Ann. Oper. Res. 253 (1), 519–543.
Papapanagiotou, Vassilis, Weyland, Dennis Alexander, Montemanni, Roberto, Gambardella, Luca Maria. 2013. A sampling-based approximation of the objective function of the orienteering problem with stochastic travel and service times. In: Proceedings of 5th International Conference on Applied Operational Research, Lecture Notes in Management Science.
Papapanagiotou, Vassilis, Montemanni, Roberto, Gambardella, Luca Maria, 2014. Objective function evaluation methods for the orienteering problem with stochastic travel and service times. J. Appl. Oper. Res. Int. 6 (1), 16–29.
Papapanagiotou, Vassilis, Montemanni, Roberto, Gambardella, Luca Maria, 2015a. The orienteering problem with stochastic travel and service. Times new approaches to sampling-based objective function evaluation. In: Proceedings of International Conference on Computational Mathematics, Computational Geometry & Statistics (CMCGS). Global Science and Technology Forum.
Papapanagiotou, Vassilis, Montemanni, Roberto, Gambardella, Luca M., 2015b. Hybrid sampling-based evaluators for the orienteering problem with stochastic travel and service times. J. Traffic Logist. Eng. 3 (2).
Papapanagiotou, Vassilis, Montemanni, Roberto, Gambardella, Luca Maria, 2016. Sampling-based objective function evaluation techniques for the orienteering problem with stochastic travel and service times. In: Operations Research Proceedings 2014. Springer International Publishing, pp. 445–450.
Pillac, Victor, Gendreau, Michel, Guéret, Christelle, Medaglia, Andrés L., 2013. A review of dynamic vehicle routing problems. Eur. J. Oper. Res. 225 (1), 1–11.
Righini, Giovanni, Salani, Matteo, 2006. Dynamic programming for the orienteering problem with time windows. Note del Polo-Ricerca 91.
Righini, Giovanni, Salani, Matteo, 2009. Decremental state space relaxation strategies and initialization heuristics for solving the orienteering problem with time windows with dynamic programming. Comput. Oper. Res. 36 (4), 1191–1203.
Schilde, Michael, Doerner, Karl F., Hartl, Richard F., Kiechle, Guenter, 2009. Metaheuristics for the bi-objective orienteering problem. Swarm Intell. 3 (3), 179–201.
Sevkli, Zülal, Sevilgen, F. Erdoğan, 2006. Variable neighborhood search for the orienteering problem. In: International Symposium on Computer and Information Sciences. Springer, Berlin Heidelberg.
Souffriau, Wouter, Vansteenwegen, Pieter, Berghe, Greet Vanden, Oudheusden, Dirk Van, 2010. A path relinking approach for the team orienteering problem. Comput. Oper. Res. 37 (11), 1853–1859.
Tang, Hao, Miller-Hooks, Elise, 2005. Algorithms for a stochastic selective travelling salesperson problem. J. Operat. Res. Soc. 439–452.
Tarantilis, Christos D., Stavropoulou, Foteini., Repoussis, Panagiotis P., 2013. The capacitated team orienteering problem: a bi-level filter-and-fan method. Eur. J. Oper. Res. 224 (1), 65–78.
Tasgetiren, M. Fatih, Smith, Alice E., 2000. A genetic algorithm for the orienteering problem. In: Proceedings of the 2000 Congress on Evolutionary Computation. vol. 2 IEEE.
Thomadsen, Tommy, Stidsen, Thomas K., 2003. The Quadratic Selective Travelling Salesman Problem.
Tricoire, Fabien, Romauch, Martin, Doerner, Karl F., Hartl, Richard F., 2010. Heuristics for the multi-period orienteering problem with multiple time windows. Comput. Oper. Res. 37 (2), 351–367.
Tsiligirides, Theodore, 1984. Heuristic methods applied to orienteering. J. Operat. Res. Soc. 797–809.
Vansteenwegen, Pieter, Van Oudheusden, Dirk, 2007. The mobile tourist guide: an OR opportunity. OR Insight 20 (3), 21–27.
Vansteenwegen, Pieter, Souffriau, Wouter, Berghe, Greet Vanden, Oudheusden, Dirk Van, 2009. Iterated local search for the team orienteering problem with time windows. Comput. Oper. Res. 36 (12), 3281–3290.
Vansteenwegen, Pieter, Souffriau, Wouter, Oudheusden, Dirk Van, 2011. The orienteering problem: a survey. Eur. J. Operat. Res. 209 (1), 1–10.
Varakantham, Pradeep, Kumar, Akshat, 2013. Optimization approaches for solving chance constrained stochastic orienteering problems. In: International Conference on Algorithmic Decision Theory. Springer, Berlin Heidelberg.
Varakantham, Pradeep, Kumar, Akshat, Lau, Hoong Chuin, Yeoh, William, 2017. Risk-sensitive stochastic orienteering problems for trip optimization in urban environments. ACM Trans. Intell. Syst. Technol.
Verbeeck, Cédric, 2016. Optimizing Practical Orienteering Problems with Stochastic Time-dependent Travel Times: Towards Congestion Free Routes (Dissertation). Ghent University.
Verbeeck, Cédric, Sörensen, Kenneth, Aghezzaf, E.-H., Vansteenwegen, Pieter, 2014a. A fast solution method for the time-dependent orienteering problem. Eur. J. Oper. Res. 236 (2), 419–432.
Verbeeck, Cédric, Vansteenwegen, Pieter, Aghezzaf, El-Houssaine, 2014b. The orienteering problem with time-dependent stochastic travel times. Verolog 2014.
Wang, Qiwen, Sun, Xiaoyun, Golden, Bruce L., Jia, Jiyou, 1995. Using artificial neural networks to solve the orienteering problem. Ann. Oper. Res. 61 (1), 111–120.
Wang, Xia, Golden, Bruce L., Wasil, Edward A., 2008. Using a genetic algorithm to solve the generalized orienteering problem. In: The Vehicle Routing Problem: Latest Advances and New Challenges. Springer, US, pp. 263–274.
Wörndl, Wolfgang, Hefele, Alexander, Herzog, Daniel, 2017. Recommending a sequence of interesting places for tourist trips. Inf. Technol. Tourism 17 (1), 31–54.
Zhang, Mengying, Qin, Jin, Yugang, Yu, Liang, Liang, 2016. Traveling salesman problems with profits and stochastic customers. Int. Trans. Operat. Res.