Les Cahiers du GERAD ISSN: 0711–2440

Team orienteering with time-varying profit

Q. Yu, Y. Adulyasak,
L.M. Rousseau, N. Zhu, S. Ma

G–2019–74

October 2019

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2019
– Bibliothèque et Archives Canada, 2019

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2019
– Library and Archives Canada, 2019

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Team orienteering with time-varying profit

Qinxiao Yu [a, c]

Yossiri Adulyasak [b]

Louis-Martin Rousseau [c]

Ning Zhu [*, a]

Shoufeng Ma [a]

[a] *College of Management and Economics, Tianjin University, Tianjin 300072, China*

[b] *GERAD & HEC Montréal, Montréal (Québec), Canada, H3T 2A7*

[c] *Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

[*] *Corresponding author*

yuqinxiao@tju.edu.cn
yossiri.adulyasak@hec.ca
louis-martin.rousseau@polymtl.ca
zhuning@tju.edu.cn
sfma@tju.edu.cn

October 2019
Les Cahiers du GERAD
G–2019–74

Abstract: This paper studies the team orienteering problem, where the arrival time and service time affect the collection of profits. Such interactions result in a non-concave profit function. This problem integrates the aspect of time scheduling into team orienteering and could be used in urban search and rescue or tourist trip planning. We formulate a mixed integer non-concave programming model for it and propose a Benders branch-and-cut algorithm, along with valid inequalities for tightening the upper bound. To solve it more effectively, we introduce a hybrid heuristic that integrates a modified coordinate search (MCS) into an iterated local search. Computational results show that valid inequalities significantly reduce the optimality gap and the proposed exact method is capable of solving instances where the mixed integer nonlinear programming solver SCIP fails in finding an optimal solution. In addition, the proposed MCS algorithm is highly efficient compared to other benchmark approaches whereas the hybrid heuristic is proven to be effective in finding high-quality solutions within short computing times. We also demonstrate the performance of the heuristic with the MCS using the instances with up to 100 customers.

Keywords: Team orienteering, routing and scheduling, mixed integer nonconvex programming, benders branch-and-cut, hybrid heuristic

Acknowledgments: The first author would like to acknowledge that this work was supported by the China Scholarship Council (CSC) scholarship and the National Natural Science Foundation of China [grant number 71431005, 71971154].

1 Introduction

The orienteering problem (OP) is a classic NP-hard combinatorial optimization problem. Since the introduction of the OP in Tsiligirides (1984) and Golden et al. (1987), several variants of OP have been addressed in the literature. The class of OP traditionally involves finding a single route, maximizing the total profits collected from the chosen customers, and respecting the given time duration. The team orienteering problem (TOP) is a natural extension of OP, as it generalizes OP to multiple routes (Chao et al. 1996). In the vast majority of OP and its variants, the profit of each customer is fixed and completely collected within the given service time. We refer the reader to the surveys on the family of OP and their applications given by Vansteenwegen et al. (2011), Gunawan et al. (2016) and Vansteenwegen and Gunawan (2019).

Our paper focuses on a variant of the TOP where the profit collected from a visited customer is time-varying. In particular, we assume that (i) the profit associated with each customer will decrease with time: the later the vehicle arrives, the less profit will remain; and (ii) the amount of profit collected increases with service time. These two assumptions result in a trade-off between the gains and losses from different customers. The aim of our problem is to determine a set of routes to visit a subset of customers, as well as to decide the duration of service time spent on each visited customer, so as to maximize the total collected profit within a given time limit. We call this problem the TOP with service and arrival time-varying profit (TOP-TTP).

The TOP-TTP is motivated by search and rescue (SAR) operations during the first 72 hours after an earthquake occurs, which is critical to ensure a high survival rate. Because numerous sites can be destroyed and people can be trapped in these affected areas, survival rates decline rapidly. With a limited number of SAR teams, it is simply impossible to assign a team to every affected site when multiple sites are involved. From a single site standpoint, the earlier the SAR team arrives and the longer the search and rescue phase continues, the larger the probability will be of finding survivors. However, from the whole area standpoint, when the number of SAR teams is insufficient, staying at one site longer unfortunately decreases the probability of survival at other sites . Thus, decision makers have to make an extremely difficult decision as to how to allocate and schedule SAR teams where people's lives are at risk. Whlie, the SAR team assignment and scheduling problem are well studied (Chen and Miller-Hooks 2012, Wex et al. 2014, Schryen et al. 2015, Nayeri et al. 2018, Rauchecker and Schryen 2019), only Chen and Miller-Hooks (2012) considered maximizing the total number of survivors and took into account that the likelihood of survival diminishes over time. To the best of our knowledge, no article has considered the duration of service time as decision variables. Another application arises in the optimal tourist routing problem, in which a tourist plans to maximize the profits by selecting some attractions and scheduling an appropriate amount of time to visit sites such as museums, theatres, and historical sites. Visiting a site generates a profit, but the profit at some sites is time-varying to avoid crowds, thus tourists may prefer to stay longer at some particular sites to ensure a better experience.

Two key difficulties arise when we jointly optimize the duration of service time with routing decision in the TOP-TTP: (i) the design of routes is challenging due to the interaction of routing decision and service time decision. The arrival time of a customer relies on the service times of its predecessors. This dependant relationship makes the problem difficult to decompose; and (ii) the profit is generally affected by service time or arrival time in a nonlinear way. Taking both of them into consideration leads to a complicated profit function which is non-concave. The contribution of this paper can be summarized as follows: (i) a non-concave MINLP model is formulated for the TOP-TTP; (ii) we propose a Benders branch-and-cut approach to address this problem, with two sets of valid inequalities to tighten the bound of this approach; (iii) we design a modified coordinate search method which can solve the service time allocation efficiently, and embed it into an iterated local search to make a hybrid heuristic algorithm for the TOP-TTP; and, finally, (iv) numerical computational results show that the proposed exact algorithm significantly outperforms a standard MINLP solver and generates tight bounds for the instances with up to 25 customers. Compared with the exact one, the heuristic algorithm provides competitive solutions to the instances with up to 100 customers in much shorter computing times.

The rest of this paper is structured as follows: a literature review of related topics is provided in Section 2; the definition of TOP-TTP and its mathematical formulation are formally described in Section 3; then, the details of the Benders branch-and-cut approach are discussed in Section 4; Section 5 presents the heuristic algorithm, which is then followed by the computational experiments in Section 6, ending with our conclusion in Section 7.

2 Literature review

2.1 (T)OP with time-varying problem

OP and its variants have attracted a lot of attention in the last few decades. While most are restricted to fixed profit, work that considers time-varying profit appears to be sparse. We roughly classify them into two streams: (i) the arrival time-varying profit (OPATP), in which profit depends on the arrival time; and (ii) the service time-varying profit (OPSTP), in which profit depends on the service time. A summary of papers on (T)OP with time-varying profits is given in Table 1.

Table 1: Summary of papers on orienteering problem with time-varying profit

Authors	Problem			Model	Solution
	#vehicle	arrival time	service time		
Erkut and Zhang (1996)	single	✓		MILP	branch-and-bound, a penalty-based heuristic
Tang et al. (2007)	multiple	✓		IP	an adaptive memory tabu search
Ekici and Retharekar (2013)	multiple	✓		MILP	a cluster-and-route heuristic algorithm
Afsar and Labadie (2013)	multiple	✓		MILP	column generation, an evolutionary local search
Erdoğan and Laporte (2013)	single		✓	MINLP	branch-and-cut
Pietz and Royset (2013)	single		✓	MINLP	branch-and-bound with relaxation problem for tight bounds
Yu et al. (2015)	single		✓	MINLP	piecewise linearization, branch-and-bound
Yu et al. (2019)	single		✓	MINLP	a tabu search
Gunawan et al. (2018)	multiple		✓	MIP	an iterated local search
This research	multiple	✓	✓	MINLP	Benders branch-and-cut, adaptive iterated local search

Erkut and Zhang (1996) were the first to introduce the OPATP in which a single vehicle has to visit a set of customers with linearly decreasing profits over arrival time. They proposed a branch-and-bound algorithm and a greedy heuristic algorithm for solving the OPATP. Tang et al. (2007) considered a multi-vehicle and multi-tour version of the OPATP and set a piecewise linear function to represent the relationship between profit and arrival time, while proposing an adaptive memory tabu search to solve it. Ekici and Retharekar (2013) extended the OPATP to a multi-vehicle version with linearly decreasing profit function, and designed a two-phase heuristic algorithm. Specifically, they first clustered the customers using k-means, then solved the routing problem for each cluster. Afsar and Labadie (2013) also studied the same problem as Ekici and Retharekar (2013), using column generation and evolutionary local search methods to calculate the lower and upper bounds of the OPATP, respectively.

Unlike OPATP, the OPSTPs need to determine not only the route, but also the appropriate service time spent on each visited customer. First introduced in Erdoğan and Laporte (2013), they discussed the collected profit which increased over service time in a convex and a concave way and presented the linearization cuts for these two cases separately. They also proposed a branch-and-cut algorithm for solving the linear models. Pietz and Royset (2013) studied OPSTP with concave increasing profit function over service time and proposed a partial path relaxation approach to give tight bound, then they presented a branch-and-bound algorithm. Yu et al. (2015) incorporated the OPSTP model into a tourist planning problem, in which the profit could be an arbitrary function with service time. They employed a piecewise linear function to approximate the nonlinear profit function and used a branch-and-bound algorithm. Yu et al. (2019) proposed a two-phase matheuristic algorithm that consists of a tabu search method and a nonlinear programming technique to solve the OPSTP with concave increasing profit effectively. Gunawan et al. (2018) extended OPSTP to multiple vehicles, and supposed a customer could be visited more than once, with each visit spending a fixed amount of time to collect a certain percentage of the profit. They proposed an integer programming model and an algorithm based on iterated local search to solve the problem.

To the best of our knowledge, no work in the literature considers the simultaneous effect of both service time and arrival time on the profit. More specifically, the service time at one customer will directly affect the result of profit attributed to this customer. But it will also affect the arrival times at subsequent customers. Thus, the TOP-TTP is an integration of time scheduling and routing problem that is highly complex.

2.2 Non-convexity in vehicle routing problems

The vehicle routing problem (VRP) attracts a lot of attentions from both academics and practitioners. However, we find that a relatively small number of studies involve the non-convexity feature in the context of VRP. As non-convexity makes the VRP, which is already NP-hard, more intractable to solve. To the best of our knowledge, very few articles focus on developing exact methods for the non-convex VPR, while the approximate methods are more common. Zhong and Aghezzaf (2011) studied the single-vehicle cyclic inventory routing problem (CIRP), in which the customer has a recurring demand, and the aim is to find a cyclic distribution plan consisting of the route and its cycle time, so as to maximize the collected rewards from the visited customers while minimizing transportation and inventory costs. This results in a MINLP with a non-convex objective function. They proposed a steepest decent algorithm combining with DC-programming for speeding up. By gradually covering the whole domain of the cycle time, they could solve the single-vehicle CIRP optimally. The proposed algorithm is capable of solving instances with up to 15 customers. Camm et al. (2017) analyzed a VRP where vehicles carry people and considered how long individuals are riding in the vehicles. The objective is to minimize the distance traveled by vehicles, weighted by the number of passengers on the routes. This results in a non-convex objective function containing the product terms of decision variables. They reformulated the MINLP model as an integer linear programming (ILP) using auxiliary variables and developed a branch-price-and-cut algorithm for the ILP. The ILP models are tested on a bus shuttle service data with 25 locations and 3 vehicles.

The following papers proposed approximate methods. Coutinho et al. (2018) considered the glider routing problem to survey a set of locations. During the flight, the trajectory of the glider must satisfy flight dynamics which are non-convex in nature. The authors convert the non-convex MINLP to a mixed integer second-order core programming problem by linearising the flight dynamics. A commercial solver is employed to produce solutions and results are provided for instances with up to 10 locations and 3 gliders. Lefever et al. (2016) also studied the single-vehicle CIRP, and modified the non-convex MINLP so that its continuous relaxation is a convex problem. Then a branch-and-bound algorithm is applied to the continuous relaxation and tested on the instances with up to 67 customers. Ibaraki et al. (2005) proposed local search algorithms for the VRP with time-window where the time-window constraint is treated as a penalty function which can be non-convex. A dynamic programming method is integrated to compute the optimal start times of services. Chen et al. (2009) studied a production scheduling and routing problem for perishable goods, in which the goods deteriorate rapidly and continuously once they are produced.The demands at retailers are stochastic. They considered an MINLP with the non-convex objective function. A heuristic algorithm integrated with a direct search algorithm is proposed and computational results are presented on instances with up to 100 retailers.

Chitsaz et al. (2016) proposed a two-phase heuristic algorithm for the CIRP with multiple vehicles. They decomposed the original problem into routing and scheduling subproblems, and developed two heuristics, one is based on the steepest descent and the other is based on building vehicle schedules day by day, for the routing and scheduling, respectively. In these papers, the maximum size of the problems is 25 nodes for exact approaches and 200 nodes for the heuristic approaches.

3 Mathematical model for the TOP-TTP

This section presents a mathematical formulation for TOP-TTP. Let $G = (N, A)$ be a complete directed graph, where the node set is $N = \{0, 1, \ldots, n+1\}$ and arc set $A = \{(i, j) | i, j \in N, i \neq j\}$. Node 0 and $n+1$ correspond to the starting and ending depot without profit. Vehicles depart from the starting depot and return to the ending depot. While $\hat{N} = \{1, \ldots, n\}$ represents the set of profitable nodes, we call them customers. Each customer i has an ideal profit p_i and a profit function $f_i(s_i, a_i)$ varying with the duration of service time s_i and the arrival time a_i. We assume that $f(s_i, a_i) = (p_i - d_i a_i)(1 - e^{-\beta_i s_i})$, in which (i) the ideal profit p_i decreases with arrival time a_i linearly, the delay ratio is d_i, so the deadline of visiting customer i is known as $D_i = p_i/d_i$. If a vehicle arrives after the deadline, then the profit vanishes, i.e., $f_i(s_i, D_i) = 0$. (ii) the collected profit increases with the duration of service time s_i concavely, β_i is the profit collecting rate and no profit is obtained when service time is zero, i.e., $f_i(0, a_i) = 0$. The curve of the profit function is shown in Figure 1. We notice that the region near the zero profit is non-concave.

Proposition 1 *The function $f_i(s_i, a_i) = (p_i - d_i a_i)(1 - e^{-\beta_i s_i})$ is non-concave for $s_i \geq 0$ and $0 \leq a_i \leq D_i$.*

The proof is immediate: f_i is continuous and twice differential, its Hessian matrix H_i can be calculated as

$$H_i = \begin{bmatrix} -\beta_i^2(p_i - d_i a_i)e^{-\beta_i s_i} & -\beta_i d_i e^{-\beta_i s_i} \\ -\beta_i d_i e^{-\beta_i s_i} & 0 \end{bmatrix}$$

Since $|H_i|_1 = -\beta_i^2(p_i - d_i a_i)e^{-\beta_i s_i} \leq 0$, and $|H_i|_2 = -(\beta_i d_i e^{-\beta_i s_i})^2 \leq 0$, so that H_i is not negative semi-definite, then f_i is non-concave. □

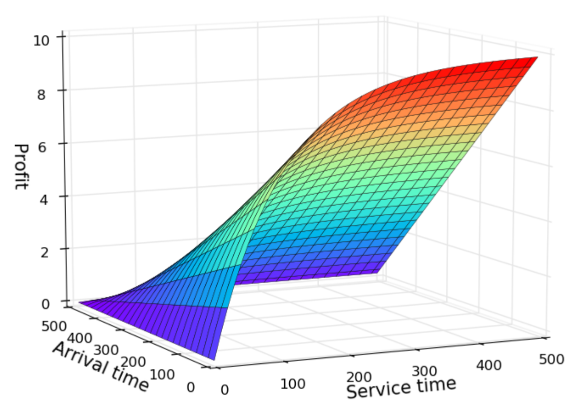

Figure 1: Profit of service time (s_i) and arrival time (a_i)

We consider a set of identical vehicles V dispatched from the starting depot to serve certain customers and return to the ending depot before the decision horizon $T_{\max}$. Each customer is allowed, at most, one visit within $T_{\max}$. Once a vehicle starts serving the customer i, the serving process cannot be interrupted. Traversing any pair of customer i and j costs t_{ij} in travel time. To maximize the overall profits, we need to determine three sets of decisions: assigning customers to vehicles; sequencing those customers; and allocating service time to each selected customer. Thus, we design the following decision variables:

- $x_{i,j,v} = 1$ if vehicle v travels directly from customer i to j and 0 otherwise.
- $y_{i,v} = 1$ if customer i is visited by vehicle v and 0 otherwise. It is an auxiliary decision variable that is uniquely determined by variables x_{ijv}.
- a_i the arrival time at customer i.
- s_i the amount of service time spend at customer i.

Then the TOP-TTP can be formulated as follows:

$$\text{[P]: } \max \sum_{i=1}^{n} (p_i - d_i a_i)(1 - e^{-\beta_i s_i}) \tag{1}$$

$$\text{s.t.} \sum_{v=1}^{|V|} \sum_{j=1}^{n} x_{0,j,v} = |V| \tag{2}$$

$$\sum_{v=1}^{|V|} \sum_{i=1}^{n} x_{i,n+1,v} = |V| \tag{3}$$

$$\sum_{i=0, i \neq k}^{n} x_{i,k,v} = \sum_{j=1, j \neq k}^{n+1} x_{k,j,v} \qquad \forall\ k \in \hat{N}, v \in V \tag{4}$$

$$\sum_{j=1, j \neq k}^{n+1} x_{k,j,v} = y_{k,v} \qquad \forall\ k \in \hat{N}, v \in V \tag{5}$$

$$\sum_{v=1}^{|V|} y_{i,v} \leq 1 \qquad \forall\ i \in \hat{N} \tag{6}$$

$$a_i + t_{ij} + s_i - a_j \leq M(1 - x_{i,j,v}) \qquad \forall\ i, j \in N, v \in V \tag{7}$$

$$a_i \leq D_i \sum_{v=1}^{|V|} y_{i,v} \qquad \forall i \in \hat{N} \tag{8}$$

$$s_i \leq T_{\max} \sum_{v=1}^{|V|} y_{i,v} \qquad \forall i \in \hat{N} \tag{9}$$

$$a_{n+1} \leq T_{\max} \tag{10}$$

$$x_{i,j,v} \in \{0, 1\} \qquad \forall\ i, j \in N, v \in V \tag{11}$$

$$y_{i,v} \in \{0, 1\} \qquad \forall\ i \in \hat{N}, v \in V \tag{12}$$

$$s_i, a_i \geq 0 \qquad \forall\ i \in N \tag{13}$$

The objective function (1) maximizes the profit collected by the vehicles. Constraints (2) and (3) ensure that exactly $|V|$ vehicles depart from the start depot and return to the end depot. Constraints (4) are the flow conservation for each customer. Constraints (5) couple routing variables $x_{i,k,v}$ with indicator variable $y_{k,v}$. Constraints (6) guarantee that at most one vehicle visits each potential customer. Constraints (14) are lexicographic ordering constraints to avoid the symmetries due to the identical vehicles. Constraints (7) couple routing decisions with the time scheduling, where M is a large positive constant. If $x_{i,j,v} = 1$, i.e., vehicle v visits customer j directly after departing from customer i, then the arrival time at customer j must be equal or larger to the arrival time at customer i plus the service time at customer i and the travel time between customers i and j. Constraints (8) ensure that the vehicle must arrive at customer i before its deadline if it is visited by a vehicle. Otherwise, the arrival time is zero. Constraints (9) force the service time not exceeding $T_{\max}$ for the visited customers, while staying at zero for the unvisited costumers. Constraint (10) ensures that the time horizon is at most $T_{\max}$. Finally, constraints (11)–(13) show the domain of the decision variables.

The fact that vehicles are identical in our problem leads to a symmetry issue. Among $|V|$ used vehicles, there are $|V|!$ options to reshuffle the routes that are assigned to each vehicle. To tackle these symmetries, we add the following lexicographic ordering constraints (which could significantly improve algorithmic performance in routing problems as shown in Adulyasak et al. (2014)) to the model [P]:

$$\sum_{i=1}^{j} 2^{(j-i)} y_{i,v} \geq \sum_{i=1}^{j} 2^{(j-i)} y_{i,v+1}, \quad \forall j \in \hat{N}, \forall 1 \leq v \leq |V| - 1 \tag{14}$$

To better understand the model and the interaction of the routing and service time decisions, Figure 2 presents a numerical example illustrating the solutions to two instances with 9 customers and 3 vehicles, while the parameter $T_{\max}$ is set to 348 and 464, respectively. In this figure, each customer is represented by a circle. The size of a circle indicates the ideal profit p_i, meaning the bigger the circle, the larger ideal profit the customer has. And the infill density describes the delay ratio of the ideal profit to the arrival time, meaning the lower the density is, the slower the ideal profit decreases with the arrival time. According to the solutions, the routes for all the vehicles, together with the collected profit, the service time and arrival time of every served customer, are presented. We notice that a significant change occurs on the route when $T_{\max}$ increases from 348 to 464, that is, customers with small ideal profits like customers 2, 6 and 9, are served earlier (as shown in Figure 2(b)) than that in Figure 2(a). The reason for this is their profits vanish earlier due to the small profit, thus, the routes are reorganized to balance the gains and losses of different customers, so as to find a better solution. The non-concave profit function captures the trade-off between the service time and arrival time, however, it makes the model [P] difficult to solve with the off-the-shelf MINLP solver. The results in section 6.2 showed that SCIP (Vigerske and Gleixner 2018), which is an MINLP solver, failed to solve most instances optimally in 2 hours when the number of customers is more than 6.

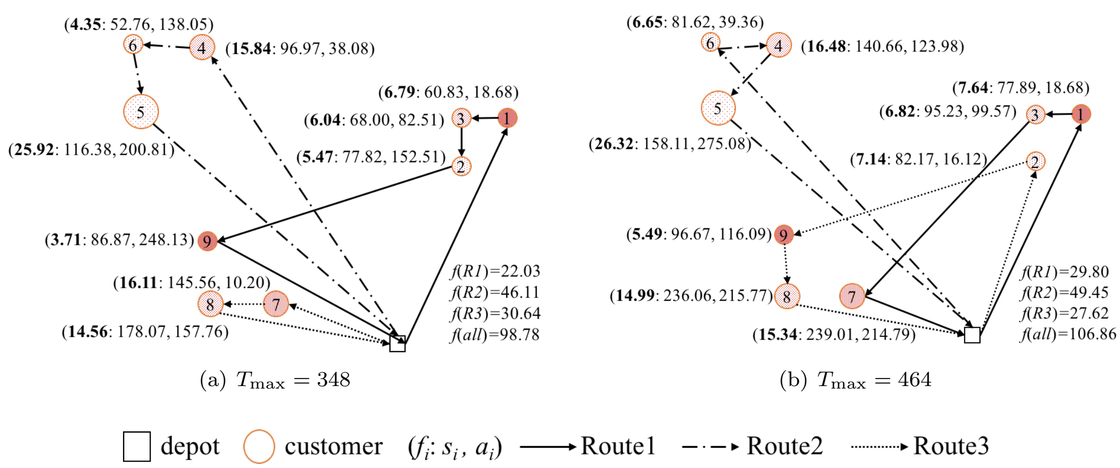

Figure 2: Example of solutions to two instances with different $T_{\max}$.

4 A tailored benders branch-and-cut algorithm

We now introduce the Benders branch-and-cut algorithm based on Benders decomposition (BD) to solve the TOP-TTP. In our study, we decompose the TOP-TTP into two distinct problems: (i) Benders master problem that determines the subset of customers that should be visited, which vehicle should be assigned to a visited customer, and the route for these visited customers; and (ii) Benders subproblem that determines the amount of service time for customers. The Benders reformulation yields the master problem which takes the form of deterministic team orienteering problem (which can be solved efficiently by a branch-and-cut algorithm) whereas the non-concave component is projected out to the Benders subproblem.

Due to the non-concavity in our Benders subproblem, the classical Benders cuts, which take advantage of the strong duality of the subproblem, are not applicable. Some authors extend the BD to handle the situation where duality information is not available. Laporte and Louveaux (1993) considered the situation in which the Benders subproblem contains binary variables, and proposed a lower bounding functional (LBF) cut using the objective value of the subproblem to generate the optimality cut. Branch-and-check described in Thorsteinsson (2001) is a variant of BD. The authors introduced the no-good cut, which yields from the solution of subproblem, when the subproblem is infeasible to disallow this solution and others similar to it. Combinatorial Benders (CB) cut (Codato and Fischetti 2006) can be reviewed as a no-good cut to separate infeasible solutions from the master problem, which only contains combinatorial information on the integer variables. We refer to Rahmaniani et al. (2017) for a recent review of the BD and its extensions.

Inspired by the above extensions of BD, the proposed BBC takes the framework of branch-and-check, and generates the upper bounding function (UBF) cuts (which correspond to the LBF for a minimization problem in Laporte and Louveaux (1993)) and the CB cuts according to the characteristic of the TOP-TTP. More specifically, given that the master problem is a relaxation of the original problem, the solution is not guaranteed to be truly feasible to the original problem. In the subproblem, we first check whether the solution is feasible to the original problem. A CB cut is added to the master problem as a feasibility cut if the solution is infeasible. Otherwise, an UBF cut is added as an optimality cut. The master problem and subproblem are solved iteratively until the lower bound from the subproblem is equal to the upper bound from the master problem, or until all the branch-and-bound nodes are examined. In the following subsections, we describe the Benders reformulation and its solution algorithm.

4.1 Benders master problem

We identified the binary variables $(\mathbf{x}, \mathbf{y})$ as complicating variables representing the set of visited customers and routes among them. The master problem [MP] is defined as follows:

$$\text{[MP]: max} \quad \sum_{v\in V} \theta_v \tag{15}$$

s.t. Constraints (2)–(14) and (11)–(12)

$$\theta_v \leq \sum_{i\in\hat{N}} f_i(t_{0i}, T_{\max} - t_{0i} - t_{in+1}) y_{i,v}, \quad \forall v \in V \tag{16}$$

$$\sum_{i\in N}\sum_{j\in N} t_{ij} x_{i,j,v} \leq T_{\max}, \quad \forall v \in V \tag{17}$$

$$\sum_{(i,j)\in\delta(S)} x_{i,j,v} \geq 2y_{k,v}, \quad \forall S \subset N,\ |S| \geq 2,\ \forall k \in N \setminus S,\ \forall v \in V \tag{18}$$

Feasibility cuts (20) and optimality cuts (27)

$$\theta_v \geq 0, \quad \forall v \in V \tag{19}$$

In the [MP], we impose new variable θ_v to represent the profit collected by the vehicle v. Constraints (16) set an upper bound for θ_v. This upper bound is derived from an optimistic case where the sum of the maximal profits collected from all visited customers in this route, the maximal profit happens when the customer is visited as the first in the sequence and the service time is set to the maximum. Constraints (17) guarantee that the travel time of a route will not exceed $T_{\max}$. However, the route satisfying this constraint may be infeasible in the original model [P], because it may violate the deadline of some customers. Constraints (18) are the generalized subtour elimination constraints (GSECs) for the vehicle routing problem (Fischetti et al. (1998)); S denotes any subset of customers on the route; $\delta(S)$ represents the set of arcs that connect customers in S with those outside S. We can use the GSECs to replace constraints (7) because the time variables a_i and s_i are now projected out to the subproblem. The GSECs are added to the [MP] in a branch-and-cut fashion once a subtour is detected at any node in the branch-and-bound tree. The minimum s-t cut problem is employed to detect a cycle and is solved by Edmonds-Karp algorithm (Edmonds and Karp (1972)). The feasibility cuts (20) and optimality cuts (27) to be defined in Section 4.2 are appended to the [MP] once the subproblem is solved.

When a feasible solution of the [MP] is found, it provides an upper bound to the original objective value (of the maximization problem) and a set of trial routes for the selected customers (which maybe infeasible to the [P]). We then provide it to the Benders subproblem. Next, we describe approaches to tackle the subproblem and derive the feasibility and optimality cuts for the [MP].

4.2 Derivation of the subproblem

Let $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ denote the solution from the [MP], then V trial routes can be extracted corresponding to $\bar{x}_{i,j,v} = 1, \forall i, j \in N$. When the vehicle routes are fixed, the Benders subproblem can be decomposed into V single-vehicle subproblems. For the $\bar{v}$-th route, we describe it as $R^{\bar{v}} = (r_0^{\bar{v}}, r_1^{\bar{v}}, \ldots, r_h^{\bar{v}}, \ldots, r_m^{\bar{v}}, r_{m+1}^{\bar{v}})$, where $r_0^{\bar{v}}$ and $r_{m+1}^{\bar{v}}$ are the starting and ending depots. Let $\bar{a}_{r_i^{\bar{v}}}$ denote the arrival time at the visited customer $r_i^{\bar{v}}$ without considering any service time. As we remove the arrival time and service time decisions in the [MP], the given trial route $R^{\bar{v}}$ may be infeasible to the Benders subproblem, e.g., arriving at some customers too late.

The infeasibility caused by violating a deadline is easy to check for a given route. We define a customer in the given route as an unreachable customer when $\bar{a}_{r_i^{\bar{v}}} > D_{r_i^{\bar{v}}}$. If there exists any unreachable customer in the route, let $\bar{R} = \{r_1^{\bar{v}}, \ldots, r_h^{\bar{v}}\}$ representing a sequence that contains the set of customers from the first visited customer $r_1^{\bar{v}}$ to the first unreachable customer $r_h^{\bar{v}}$. Then, the following feasibility cuts can be added to the [MP] (see Codato and Fischetti (2006)):

$$\sum_{r_i^{\bar{v}} \in \bar{R} \setminus \{r_h^{\bar{v}}\}} x_{r_i^{\bar{v}}, r_{i+1}^{\bar{v}}, v} \leq h - 2, \ \forall v \in V \tag{20}$$

Every route containing the above sequence $\bar{R}$ will be infeasible, thereby forcing the number of arcs in it to be strictly smaller than $h - 1$.

When every customer is reachable in $R^{\bar{v}}$, i.e., $\bar{a}_{r_i^{\bar{v}}} \leq D_{r_i^{\bar{v}}}$ holds for $\forall r_i^{\bar{v}} \in R^{\bar{v}}$. What we need to decide is the duration of service time $s_{r_i^{\bar{v}}}$ assigned to each customer $r_i^{\bar{v}}$ so as to maximize total profits, as well as to respect the real arrival time constraints and the total service time constraint. Let $\hat{T}_{\bar{v}}$ denote the remaining total service time which equals $T_{\max} - \sum_{i,j \in N} t_{ij} \bar{x}_{i,j,\bar{v}}$, and the Benders subproblem for $\bar{v}$-th route is defined as follows:

$$\text{[SSP]: } \max \sum_{i=1}^{m} f_{r_i^{\bar{v}}}(a_{r_i^{\bar{v}}}, s_{r_i^{\bar{v}}}) \tag{21}$$

$$\text{s.t. } \sum_{i=1}^{m} s_{r_i^{\bar{v}}} \leq \hat{T}_{\bar{v}} \tag{22}$$

$$a_{r_i^{\bar{v}}} + s_{r_i^{\bar{v}}} + t_{r_i^{\bar{v}}, r_{i+1}^{\bar{v}}} = a_{r_{i+1}^{\bar{v}}} \qquad \forall i = 0, \ldots, m-1 \tag{23}$$

$$a_{r_i^{\bar{v}}} \leq D_{r_i^{\bar{v}}} \qquad \forall i = 1, \ldots, m \tag{24}$$

$$a_{r_0^{\bar{v}}}, s_{r_0^{\bar{v}}} = 0 \tag{25}$$

$$a_{r_i^{\bar{v}}}, s_{r_i^{\bar{v}}} \geq 0 \qquad \forall = 1, \ldots, m \tag{26}$$

Note that this subproblem is always feasible and its objective cannot be less than zero when every customer is reachable, because there exists a solution $\{s_{r_i^{\bar{v}}} = 0, a_{r_i^{\bar{v}}} = \bar{a}_{r_i^{\bar{v}}} | \forall i = 1, ..., m\}$ satisfying all the constraints. Solving the [SSP], we get the optimal service time scheduling $\mathbf{s}_{\bar{v}}^*$ and the optimal objective value $\hat{\theta}_{\bar{v}}$ for the given route. Similar to the LBF in Laporte and Louveaux (1993), we generate the following optimality cut for every feasible route $R_{\bar{v}}$ and add it to the [MP]:

$$\theta_{\bar{v}} \leq \hat{\theta}_{\bar{v}} + (\hat{\theta}_{\bar{v}} - \mathcal{M}_{ub}) W(\mathbf{x}) \tag{27}$$

$$W(\mathbf{x}) = \sum_{(i,j) \in R_{\bar{v}}} x_{i,j,\bar{v}} - \sum_{i \in \hat{N}} \bar{y}_{i,\bar{v}} - 1$$

where $\mathcal{M}_{ub}$ is an upper bound on the objective function of each route, here we set $\mathcal{M}_{ub} = \sum_{i\in\hat{N}} p_i$, which is the sum of the ideal profits of all customers. The cuts (27) are valid because the total number of arcs in the first part of $W(\mathbf{x})$ is at most $\sum_{i\in\hat{N}} \bar{y}_{i,\bar{v}} + 1$, $W(\mathbf{x})$ is zero for the current route, and is a negative integer for other routes. Therefore, the cuts (27) impose $\theta_{\bar{v}} \le \hat{\theta}_{\bar{v}}$ at the current route, and $\theta_{\bar{v}} \le \mathcal{M}_{ub} + z^{+}(\mathcal{M}_{ub} - \hat{\theta}_{\bar{v}})$ at other routes (z^{+} is a positive integer).

4.3 Tightening the upper bound

Because the original objective function is projected out from the Benders master problem [MP], the resulting upper bound can be weak. Recall that we assume that the collected profit decreases when both the arrival time increases and the service time decreases. If we assume the collected profit only depends on either arrival time or service time, the relaxed function then yields an upper bound of the original profit function.

More specifically, the real collected profit θ_v of the route R_v is computed as the formulation (28), where $a_{i,v}$ and $s_{i,v}$ are defined to denote the time vehicle v arrives at customer i and the duration of service time that vehicle v spends on customer i.

$$\theta_v = \sum_{i\in\hat{N}} (p_i - d_i a_{i,v})(1 - e^{-\beta_i s_{i,v}}) y_{i,v} \tag{28}$$

For the case where only arrival time is considered, we have inequality (29) because $1 - e^{-\beta_i s_{i,v}}$ is less than 1 in the equation (28). To avoid quadratic term $a_{i,v} y_{i,v}$, we add constraints (30) and (31) to force $a_{i,v}$ to be zero when $y_{i,v} = 0$. Since these two constraints only restrict the bounds of the arrival time at a visited customer, constraints (32) are required to link the arrival times between any pair of two adjacent customers, otherwise the arrival time at each visited customer will take the lower bound t_{0i} to make θ_v, resulting in a larger profit.

$$\theta_v \le \sum_{i\in\hat{N}} (p_i y_{i,v} - d_i a_{i,v}) \tag{29}$$

$$a_{i,v} \le D_i y_{i,v}, \forall i \in \hat{N} \tag{30}$$

$$a_{i,v} \ge t_{0i} y_{i,v}, \forall i \in \hat{N} \tag{31}$$

$$a_{j,v} \ge a_{i,v} + t_{ij} + s_{i,v} - \mathcal{M}(1 - x_{i,j,v}), \forall i, j \in \hat{N} \tag{32}$$

Similarly, the inequality (33) holds when we only consider the effect of service time on collecting profit. And inequalities (34) are imposed to restrict $s_{i,v}$ as well as to eliminate the complicated term $e^{-\beta_i s_{i,v}} y_{i,v}$ in inequality (33), the inequality (35) strengthens constraint (17) by incorporating the service time variables.

$$\theta_v \le \sum_{i\in\hat{N}} p_i (1 - e^{-\beta_i s_{i,v}}) \tag{33}$$

$$s_{i,v} \le (T_{\max} - t_{0i} - t_{i0}) y_{i,v}, \forall i \in \hat{N} \tag{34}$$

$$\sum_{i\in N} \sum_{j\in N} t_{ij} x_{i,j,v} + \sum_{i\in\hat{N}} s_{i,v} \le T_{\max} \tag{35}$$

We notice that the inequality (33) is concave and could be linearized by introducing auxiliary variables $w_{i,v}$, representing the profit collected by vehicle v at customer i, for all customers. Also by using the linearization inequalities mentioned in Erdoğan and Laporte (2013), we reformulate it as follows:

$$\theta_v \le \sum_{i\in\hat{N}} w_{i,v} \tag{36}$$

$$w_{i,v} \le p_i(\beta_i e^{-\beta_i s^*_{i,v}} s_{i,v} + 1 - e^{-\beta_i s^*_{i,v}} - s^*_{i,v} \beta_i e^{-\beta_i s^*_{i,v}}),\ \forall i \in \hat{N}, \forall s^*_{i,v} \in [0, T_{\max}] \tag{37}$$

Finally, the [MP] could be reformulated into the enhanced master problem [EnMP] as follows:

$$
\begin{aligned}
\text{[EnMP]: max} \quad & \sum_{v \in V} \theta_v \\
\text{s.t.} \quad & \text{Constraints (2)–(14), (11)–(12), (18) and (19)} \\
& \text{Constraints (29)–(32), (34)–(37)} \\
& w_{i,v} \geq 0,\ s_{i,v} \geq 0,\ a_{i,v} \geq 0, \quad \forall i, j \in \hat{N}, \forall v \in V
\end{aligned} \tag{38}
$$

Similar to the [MP], the [EnMP] is also solved in a branch-and-cut fashion. Particularly, the GSEC constraints (18) are added at a node in the branch-and-bound tree when a subtour is detected. The adjacent arrival times constraints (32) are added for every arc between two adjacent customers in a route which extracted from the integer solution of [EnMP] at a node when a feasible route is found, and the linearization constraint (37) is checked at every node and added only if it is violated. As in the case of [MP], when an integer solution of [EnMP] is found, we pass the vector $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ to the Benders subproblem.

5 The hybrid heuristic algorithm

In this section, we introduce the hybrid heuristic algorithm for solving our problem. Similar to the Benders decomposition, we partition the original problem into two subproblems: the routing subproblem in which routes are constructed and improved by an iterated local search (ILS); and the service time scheduling subproblem which complements every partial solution via deciding the service time at each customer. Considering a huge number of routes could be generated during ILS, quickly solving the latter subproblem is crucial for the efficiency of the entire process. Instead of using a NLP solver, here we design a heuristic algorithm modified from coordinate search. In the rest of the paper, we refer to the hybrid heuristic method as the iterated local search with modified coordinated search (ILS-MCS).

5.1 Framework of the ILS-MCS

An ILS mainly consists of local search and perturbation. At the beginning of ILS, we initialize a feasible solution $sol = \{(R^v, \mathbf{s}^v) | v = 1, \cdots, |V|\}$, then execute a number of iterations of local search. Note that the definitions of R^v and $\mathbf{s}^v$ are the same as in Section 4.2. At the beginning of each iteration, a perturbation is carried out on the current local optimal solution so as to diversify the search procedure. This is then followed by the local search, which consists of several neighbourhood structures. In every neighbourhood, a number of possible moves are applied on the current routing decisions until the best move is found. The MCS method is called to evaluate each move R in order to get the service time decision $\mathbf{s}$ and the objective value $z(R, \mathbf{s})$. We store every new route, together with its service time decision and objective value, into a pool $\mathscr{P}$. When a route needs to be evaluated, we first check the pool. As such, we avoid computing the same route repeatedly. Finally, the algorithm terminates when the best found solution has not been improved in $\mathit{\Gamma}_{\max}$ consecutive iterations. The framework of ILS-MCS is shown in Algorithm 1. We provide the details on the initialization, perturbation, and local search of ILS in Appendix A, and present how the MCS method is adapted for the service time scheduling subproblem in the following section.

5.2 The modified coordinate search for service time scheduling problem

Once a route is generated, the service time decision of this given route needs to be decided by solving the [SSP], defined in Section 4.2. Based on the character of the [SSP], we propose a heuristic method which modifies the coordinate search method (CS) and does not require any derivatives. We refer to this as the MCS in the rest of the paper. The experiments in Section 6.3.1 show that the MCS is more efficient in solving the [SSP] than other methods, including gradient-based methods and derivative-free methods.

Algorithm 1 The ILS-MCS algorithm

Input: A set of neighbourhood structures $N = \{N_l | l = 1, \dots, L\}$ and the stop criteria $\Gamma_{\max}$.
Initialize a feasible solution $sol = \{(R^v, S^v) | v = 1, \cdots, |V|\}$, let $sol^* = sol$ and $\tau = 0$.
Initialize a route pool $\mathscr{P} = \emptyset$.
while $\tau < \Gamma_{\max}$ **do**
  $sol' = \text{Perturb}(sol)$.
  for all N_l **do**
    $sol'' = LocalSearch(N_l, sol', MCS, \mathscr{P})$.
    if $f(sol'') > f(sol')$ **then** $sol' = sol''$ **end**
  end for
  if $f(sol') > f(sol)$ **then** $sol = sol'$ **end**
  if $f(sol') > f(sol^*)$ **then** $sol^* = sol'$, $\tau = 0$ **else** $\tau = \tau + 1$ **end**
end while
Output: Best found solution sol^*

The basic CS approach provides the idea that the maximization of a multivariate function can be achieved by maximizing it along a coordinated direction at a time. Iterations are cyclically executed through different coordinates until the stop criteria is satisfied (Fermi and Metropolis 1952). Although it is not a competitive optimization algorithm, it provides the logic for other advanced approaches in the subject of direct search (Audet and Hare 2017). The MCS extends the CS in two aspects: (i) the search direction is a weighted combination of all objective values changing along the corresponding coordinates; and (ii) a greedy repair procedure is executed on an infeasible solution. An overview of our MCS is presented in Algorithm 2.

Algorithm 2 The modified coordinate search

1: **Input:** Step size α and its stopping threshold $\alpha_{\min}$, the distance of a neighbor $||\vec{e_i}||$.
2: Generate a service time solution $\mathbf{s}^0 = (s_1^0, s_2^0, \dots, s_m^0)$, let $\kappa = 0$, $\mathbf{s}^* = \mathbf{s}^\kappa$.
3: **while** $\alpha > \alpha_{\min}$ **do**
4:   Construct $\mathscr{P}^\kappa = \{\mathbf{s}^\kappa + \vec{e_i} : i = 1, \dots, m\}$, calculate $\vec{v}$ as Equation (39).
5:   Compute the trail solution $\mathbf{s}' = \mathbf{s}^\kappa + \alpha\vec{d}$ as Equation (42).
6:   **for** customer v_i in the given route **do**
7:     **if** the deadline constraint is violated according to $\mathbf{s}'$ **then**
8:       $\Delta T = a_{v_i} - D_{v_i}$.
9:       **while** $\Delta T > 0$ **do**
10:         **for all** $s'_j \in \mathbf{s}', j < i$ **do**
11:           $\Delta s'_j = \min\{s'_j, \Delta T\}$, $\mathbf{s}'_j = (s'_1, s'_2, \dots, s'_j - \Delta s'_j, \dots, s'_m)$ and $\Delta F_j = F(\mathbf{s}') - F(\mathbf{s}'_j)$.
12:         **end for**
13:         Choose $k \in \arg\min_{k \in \{1,\dots,j\}} \{\Delta F_k\}$, $\Delta T = \Delta T - \Delta s'_k$, $\mathbf{s}' \leftarrow \mathbf{s}'_k$.
14:       **end while**
15:     **end if**
16:   **end for**
17:   **if** The total time budget is violated according to $\mathbf{s}'$ **then**
18:     $\Delta T = \sum_{i=1}^n s'_i - \hat{T}$.
19:     **while** $\Delta T > 0$ **do**
20:       The same as Line 10 - 13 except for all $s'_j \in \mathbf{s}'$ in Line 10.
21:     **end while**
22:   **end if**
23:   **if** The objective value is improved **then** $\mathbf{s}^* = \mathbf{s}'$, $\kappa = \kappa + 1, \mathbf{s}^\kappa = \mathbf{s}'$
24:   **else** $\alpha = \alpha/2$, $\kappa = \kappa + 1$ **end**
25: **end while**
26: **Output:** $\mathbf{s}^*$

We assume the sequence of all customers in a given route is $\hat{R} = \{r_1, r_2, \dots, r_m\}$ and the remaining total service time is $\hat{T}$. The MCS algorithm starts from an initial solution $\mathbf{s}^0$ in which each element is set to $\hat{T}/m$. At iteration κ, a set of neighbour points $\mathscr{C}^\kappa = \{\mathbf{s}^\kappa + \vec{e}_i | i = 1, \dots, m\}$ of the current point is produced, in which $\vec{e}_i$ is a vector indicating the direction along the ith coordinate axe. Then, we use the following measure to generate a search direction $\vec{v}$ of the objective function f at point $\mathbf{s}^\kappa$ in order to improve the objective value:

$$\vec{v} = \sum_{p=1}^{m} w_p \vec{e_p} \tag{39}$$

where

$$w_p = \frac{f(\mathbf{s}_p) - f(\mathbf{s}^\kappa)}{\sum_{q=1}^{m} |f(\mathbf{s}_q) - f(\mathbf{s}^\kappa)|}, \ \forall \ \mathbf{s}_p \in \mathscr{C}^\kappa \tag{40}$$

$$\vec{e_p} = \frac{\mathbf{s}_p - \mathbf{s}^\kappa}{\|\mathbf{s}_p - \mathbf{s}^\kappa\|}, \ \mathbf{s}_p \in \mathscr{C}^\kappa \tag{41}$$

Moving along this direction, we get a trail solution $\mathbf{s}'$ as

$$\mathbf{s}' = \mathbf{s}^\kappa + \alpha \vec{d} \tag{42}$$

where α is a step size and $\vec{d} = \vec{v}/\|\vec{v}\|$ is the normalized search direction. The trial solution is guaranteed to get a better objective value, but it may violate the total service constraint and/or the deadline constraints. If any violation happens, a repair will be executed on the trial solution by shortening the service time in a greedy fashion. More specifically, the amount of the violation is denoted by ΔT which is equal to $\sum_{i=1}^{m} s_i - \hat{T}$ when the total time budget constraint (22) is violated and is equal to $a_i - D_i$, if the deadline constraint (24) is violated. We first check the deadline constraint of every customer in the given route from head to tail. If the deadline of one customer is exceeded, the customers prior to it are considered as candidates. We try one candidate at a time, i.e., reduce its service time s_i in the trial solution by $\Delta s_i = \min\{\Delta T, s_i\}$; then we choose one candidate, from among all candidates, which results in the maximum objective value. We then implement the corresponding reduction on the trail solution and update the amount of violation. This procedure continues until every customer is served before its deadline. Next, we check the violation of total time budget constraint. In this case, the repair process is the same as that for the deadline violation, except we consider all customers as candidates. The trial solution becomes feasible after the above repairs. If its objective value improves, the next iteration starts; otherwise, we decrease the step size by half and continue with the current solution. The algorithm terminates when the step size is smaller than the predefined threshold.

6 Computational experiments

In this section, we present numerical experiments to test the performance of the proposed solution methods. The experiments are run on a MacOS PC with a 2.7 GHz Intel Core i5 CPU and 8 GB of memory using a single thread. The BBC method is implemented in C++ programming language, and the Benders master problem is solved by the IBM CPLEX Optimization Solver 12.8, while the NLP solver IpOpt 3.12 is used for the nonlinear Benders subproblems. All cuts and valid inequalities are added to the master problem using the callback functions available in the CPLEX. We also use SCIP optimization suite 6.0 to solve the original model [P] directly. For the heuristic algorithm ILS-MCS, we implement it in C# programming language.

6.1 Test instances

Since there is no existing benchmark for TOP-TTP, we use a subset of Solomon's instances to create our test data set. It consists of instances with cluster-located customers (C) and random-located customers (R). The number of customers n belongs to $\{6, 9, 25\}$, the number of vehicles V belongs to $\{2, 3, 4, 5\}$ and the length of time limit is equal to $T_{\max} \in \{L, 2L, 3L, 4L\}$, where L is the perimeter of the minimum bounding rectangle of all nodes in the instance, thus there are a total of 96 instances. We provide the details of these instances in Appendix B. The instances can be obtained from `https://github.com/QinxiaoYu/InstanceData`.

6.2 Computational performance of the BBC and its enhancement

In this section, we aim to test the performances of the BBC and its enhancement, which takes the [EnMP] as Benders master problem instead of the [MP]. We also employ SCIP to solve the formulation [P] directly. All algorithms are given a time limit of 2 hours. Table 2 provides the average results of the different number of customers for the three approaches. The following details are provided: the number of customers n, the number of instances # and the number of instances solved to optimality $\#O$ in each group, the optimal objective value z^*, the lower bound (also the best found objective value) z_{lb}, the upper bound z_{ub}, the percentage gap measured as $(z_{ub} - z_{lb})/z_{lb}$, and the total computing time in seconds. Boldface letters are used to indicate the best results. The results show that the enhanced

Table 2: Average computational results by three exact approaches

n	#	SCIP					BBC					Enhanced BBC				
		$\#O$	z_{lb}	z_{ub}	Gap	Time(s)	$\#O$	z_{lb}	z_{ub}	Gap	Time(s)	$\#O$	z_{lb}	z_{ub}	Gap	Time(s)
6	32	32	**54.40**	54.40	0.0%	57.9	32	**54.40**	54.40	0.0%	18.6	32	**54.40**	54.40	0.0%	2.4
9	32	4	81.64	107.85	37.8%	6743.4	8	82.92	108.14	38.2%	6458.7	24	**83.28**	86.27	3.4%	2819.6
25	32	0	142.67	364.86	205.0%	7200.0	0	146.37	364.86	175.5%	7200.0	0	**172.82**	243.61	38.3%	7200.0

BBC outperforms the other two approaches and that the performance of the BBC is competitive with SCIP. For the 6 customer instances, these three approaches could solve all instances to optimality, however the enhanced BBC is the most efficient one. This approach could solve all the instances within 3 seconds, on average, whereas BBC and SCIP spend 18.6 and 57.9 seconds, respectively. In the 9 customer instances, the enhanced BBC could solve 24 instances to optimality within 2 hours, while BBC could solve 8 instances, and SCIP could solve only 4 instances. The average computing times for the instances that solved to optimality are shorter that those reported in Table 2, which are influenced by the time limit. They are 3547.4, 4235.0 and 1359.5 seconds for SCIP, BBC and enhanced BBC, respectively. For the quality of solutions, SCIP and BBC obtain almost the same quality results on average, while the enhanced BBC provides slightly better lower bounds. Nonetheless, it decreases the upper bounds significantly, resulting in considerable decline in the gap. For the 25 customer instances, these three approaches fail to find the optimal solution within 2 hours, but the enhanced BBC provides much better lower and upper bounds compared to other approaches. This increased performance is due to the valid inequalities proposed in Section 4.3. In addition, a tight upper bound also helps speed up the search process and more nodes could be pruned during the search tree. The detailed results of all the instances are provided in Appendix C.1.

6.3 Computational performance of hybrid heuristic ILS-MCS

In this section, we first evaluate the performance of the MCS via comparison with other methods for the [SSP]. Then, we present the results obtained from the ILS-MCS and compare them with the enhanced BBC to show the effectiveness of the hybrid heuristic algorithm ILS-MCS.

6.3.1 Comparison of MCS and other methods for solving the [SSP]

To validate the performance of the MCS, we select six methods from two off-the-shelf NLP solvers (NLopt and NOMAD) beside IpOpt for comparison. The eight methods for solving [SSP] are listed in Table 3. Among these methods, IpOpt, MMA and SLSQP are gradient-based algorithms; the others belong to the derivative-free optimization methods. In addition, only the IpOpt is the exact method. We test them on a group of routes in which the number of customers varies from 2 to 30, and 5 routes are generated randomly for each given number. The maximal computing time for each method is set to 1 second. The detailed description on generating these routes, and the parameters used in each particular method, are available in Appendix C.2.

We observe that the IpOpt could reach the optimal value z^* for every route within 1 second, thus the gap between other methods and IpOpt is measured as $|z^* - z|/z^*$, where z denotes the objective value from other methods. Figure 3 and 4 present the average gaps and the average computing times of different methods for solving the [SSP] of different numbers of customers. In Figure 3, we see that IpOpt spends 20 to 30 milliseconds to solve the problem and the computing time increases slightly when the number of customers increases. It is obvious to see that IpOpt dominates NOMAD, ISRES and CS, both in quality and speed. The two gradient-based methods (MMA and SLSQP) and MCS spend far less time than IpOpt on average. With respect to quality, the gaps from MCS are very close to zero, while MMA and SLSQP perform worse than MCS on average, especially when the number of customers are small as they generate relatively large gaps. For COBYLA, it could produce near optimal solutions, but its computing times are not competitive. Thus, we conclude that MCS is the most efficient method for solving the [SSP] when speed is important. This is due to the fact that MCS requires much less computing time compared to IpOpt when the number of customers per route is not more than 25 per route; this is typically the case in practice. We thus employ the MCS in the subsequent experiments.

Table 3: Eight methods for solving [SSP]

Name	Reference	Gradient-based	Exact
Interior Point Optimizer (IpOpt)	Wächter and Biegler (2006)	Yes	Yes
Method of Moving Asymptotes (MMA) [1]	Svanberg (2002)	Yes	No
Sequential Least-Squares Quadratic Programming (SLSQP)[1]	Kraft (1994)	Yes	No
Improved Stochastic Ranking Evolution Strategy (ISRES)[1]	Runarsson and Yao (2005)	No	No
Constrained Optimization by Linear Approximations (COBYLA)[1]	Powell (1994)	No	No
Nonliear Optimization with the Mesh Adaptive Direct Search (NOMAD)[2]	Le Digabel (2011)	No	No
Coordinate Search (CS)[2]	Fermi and Metropolis (1952)	No	No
Modified Coordinate Search (MCS)	This research	No	No

[1] These methods are included in the NLopt library available on Johnson (2019).
[2] These methods are included in the blackbox optimization software NOMAD available on Abramson et al. (2019).

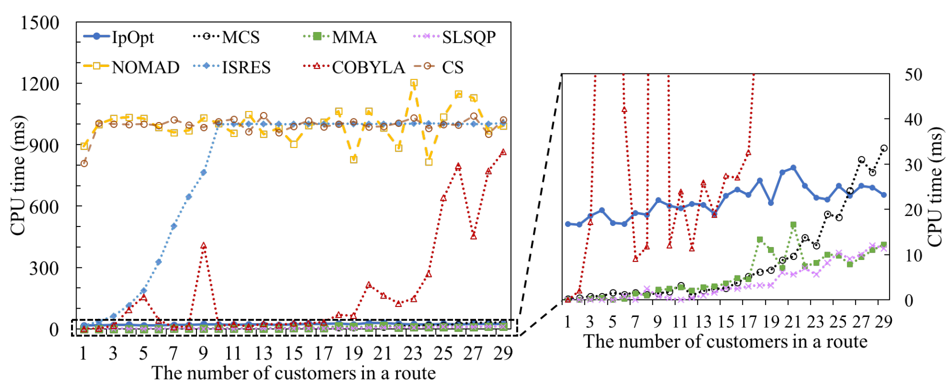

Figure 3: Average computing times (in Millisecond) by Different Approaches Solving the [SSP]

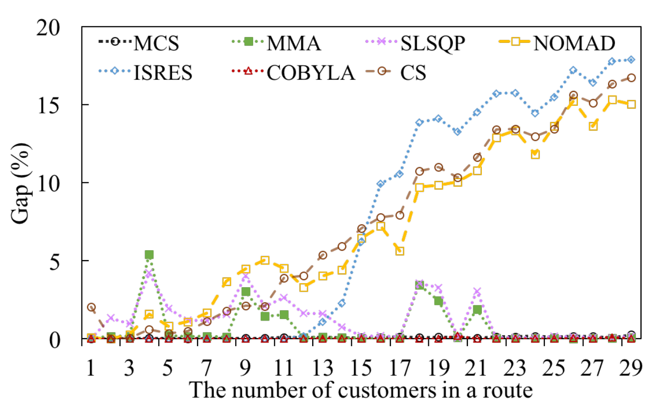

Figure 4: Average gaps by different approaches solving the [SSP]

6.3.2 Comparison of ILS-MCS and the enhanced BBC

In this section, we report the performance of the ILS-MCS and evaluate ILS with IpOpt in order to provide computational insights. For ILS-MCS and ILS-IpOpt, we set $\Gamma_{\max} = 30$ as the stopping criteria. The values of other parameters in ILS are available in Appendix A. For each instance, we run ILS-MCS and ILS-IpOpt five times, respectively. The results from these two approaches are compared with those from the enhanced BBC. We provide the average results of instances with 6, 9 and 25 customers in Table 4, the fourth column "Found Time (s)" refers to the time that the enhanced BBC found the lower bound z_{lb}. For ILS-MCS and ILS-IpOpt, we report the best, worst, and average objective values; the standard deviation among five runs; and the average total computing time, in seconds, of five runs (T.Time). The reported best objective value is guaranteed to find in T.Time. Boldface is used to indicate the best value, as well as the shortest computing time among different methods. Note that the details on the solution of each instance can be found in Appendix C.3.

From the results shown in Table 4, it is clear that ILS-MCS could generally find better solutions than the enhanced BBC, within considerably less computing time. For the instances with 6 and 9 customers, ILS-MCS also finds the optimal solutions for instances in which the enhanced BBC achieves optimality. For the computing time, ILS-MCS requires no more than 1 second to get the solution for these instances. For instances with 25 customers, ILS-MCS gets better solutions than the enhanced BBC for all the instances in this set. On average, ILS-MCS increases the best found objective value of the enhanced BBC by 6.64%. Meanwhile, the computing time of ILS-MCS is two orders of magnitude less than that of the enhanced BBC. We further remark that our heuristic method ILS-MCS is robust because the coefficient of variation is 0.14%, on average, for all the instances with up to 25 customers. From Table 4, we also see that the ILS-MCS produces solutions as good as ILS-IpOpt, however, the computing time of ILS-MCS is on average 2.74% of that of ILS-IpOpt. Thus, the MCS method could speed up the heuristic algorithm significantly.

Finally, we test ILS-MCS on the instances with 50, 75 and 100 customers, which are too large to be solved by the enhanced BBC. The average results are shown in Table 5. We report the average computing time (Avg.Time) for each run in this table, so as to compare the average objective value (Avg.Obj) with the upper bound from the exact algorithm, thus the last column "Gap" in ILS-MCS is measured as $(z_{ub} - Avg.Obj)/Avg.Obj$. We see that the enhanced BBC could only generate solutions for the instances with 50 customers, and the average gap is 504.21%, which reveals that the bounds are generally loose. Comparing this to the exact method, ILS-MCS produces significantly better solutions, with the average gap at 30.75%. Meanwhile, the computing time of ILS-MCS is 52.82 seconds on average. When the number of customers is more than 50, the enhanced BBC fails to solve any instance within 2 hours. The average gaps generated by ILS-MCS are 37.44% and 35.37%, for 75 and 100 customers, respectively, and its computing times are 345.2 and 586.2 seconds. The coefficient of variation of the ILS-MCS is on average 0.17%, for all instances with up to 100 customers. The details of the results obtained by ILS-MCS for these instances can be found in Appendix C.2.

Table 4: Average results comparing the exact and heuristic methods on small instances

	Enhanced BBC			ILS-MCS					ILS-IpOpt				
Instance	z_{lb}	Gap	Found Time(s)	Best Obj	Worst Obj	Avg. Obj	Std. Dev	T. Time(s)	Best Obj	Worst Obj	Avg. Obj	Std. Dev	T. Time(s)
6	**54.40**	0.00%	1.12	**54.40**	54.39	54.38	0.01	**0.30**	**54.40**	54.34	54.38	0.02	8.35
9	**83.28**	3.44%	749.07	**83.28**	83.23	83.26	0.02	**1.00**	83.26	83.09	83.21	0.07	68.05
25	172.82	38.28%	3914.08	**184.84**	182.93	184.29	0.78	**59.80**	**184.84**	184.31	184.64	0.21	2144.40

Table 5: Average results comparing the exact and heuristic methods on large instances

Instance	Enhanced BBC			ILS-MCS					
	z_{lb}	z_{ub}	Gap	Best Obj	Worst Obj	Avg.Obj	Std.Dev	Avg.Time(s)	Gap
50	97.82	396.37	504.21%	294.89	293.71	294.41	0.44	52.82	30.75%
75	-	612.91	-	428.71	427.02	428.06	0.64	345.19	37.44%
100	-	677.54	-	485.47	482.78	484.41	1.00	586.23	35.37%

- indicates the enhanced BBC did not find the lower bound within 2 hours.

7 Conclusions

This paper studies the team orienteering problem with service and arrival time-varying profit. Due to the interaction between the service and arrival times, the formulation of this problem becomes a mixed integer nonlinear programming (MINLP) maximizing a non-concave objective function. To solve this problem, we proposed a Benders branch-and-cut algorithm where the non-concave function is projected out to the Benders subproblem. Two sets of valid inequalities were proposed to strengthen the upper bound. The results show that the enhanced version of the BBC significantly outperformed the standard MINLP solver. To solve large instances up to 100 customers, we also developed a hybrid heuristic method ILS-MCS, which is effective in finding good-quality solutions within a few seconds.

We envision several directions for future research. As such, one is to consider the nonlinear or piecewise-linear relationship between profit and arrival time. Another direction would be to investigate machine learning techniques to select a subset of potential customers beforehand, so as to solve a variant of the vehicle routing problem, other than the team orienteering problem, which could reduce the solution space substantially.

Appendices

A Detailed description of the ILS-MCS algorithm

The ILS procedure discussed in Section 5 of the article consists of three main phases, i.e., initialization, perturbation and local search.

A.1 Initialization phase

Considering that more than one factors have an impact on collecting profit, it is difficult to deal with the visiting priority under these interactive factors. For example, a customer with a large profit may attract the visit. However, if it requires a long service time, delaying the visit to succeeding customers may result in higher loss of profit. In this case, it is possible that prioritizing a visit to a nearer customer with a larger delay ratio may not be as profitable as a visit to a further one with a smaller delay ratio. To take into account such different contributing factors, we assign a score to each customer. More specifically, a customer with a larger ideal profit, smaller delay rate, steeper collecting coefficient and shorter travel time required to traverse will have a higher priority to be visited first. We compute the score of customer i as follows:

$$\zeta_{iv} = \frac{p_i \beta_i}{d_i \Delta t_{iv}} \tag{43}$$

where p_i, d_i and β_i are ideal profit, delay rate and collecting rate, respectively. Δt_{iv} is a route-specific value which equals the incremental travel time when customer i is added to the tail of route k. Let r_v denote the last customer visited by vehicle v, we have $\Delta t_{iv} = t_{r_v,i} + t_{i,n+1} - t_{r_v,n+1}$.

To construct the initial solution, we employ a parallel insertion method which builds $|V|$ routes simultaneously. Firstly, $|V|$ routes that only consist of the starting depot 0 and ending depot $n+1$ are initialized. Let τ_v denote the departure time from the last customer r_v in route R_v and $\hat{V}$ denote a set of incomplete routes. Then, we try to assign the customer i with the highest score ζ_{iv} to the associating route R_v. If the insertion does not satisfy the time limit constraint even without considering the service time of customer i, the associating route R_v is reviewed as a complete route and removed from the incomplete set $\hat{V}$. Otherwise, the insertion procedure is called. Recall that service time must be spent to obtain profit, thus, it is plausible to assume that the service time spent on the visiting customer is larger than zero. Thus, we set the amount of service time s_i at each customer i is as long as to collect 50% of profit when the remaining time is sufficient, otherwise, the amount of service time is the remaining time. We provide the pseudocode in Algorithm 3.

Algorithm 3 Initialization

Input: the set of customers $\hat{N} = \{1, \dots, n\}$, the decision horizon $T_{\max}$, the number of vehicles $|V|$
Let $sol = \{(R^v, S^v) | R^v = \{0, n+1\}, S^v = \{0, 0\}, \forall v \in V\}$
Set $r_v \leftarrow 0$, $\tau_v \leftarrow 0$ for every route $v \in V$ and $\hat{V} = \{1, \dots, |V|\}$
while $\hat{V} \neq \emptyset$ **do**
    for all $i \in \hat{N}$ **do**
        for all $v \in V$ **do**
            $\zeta_{iv} = \frac{p_i \beta_i}{d_i \Delta t_{iv}}$.
        end for
    end for
    Choose $(i^*, v^*) \leftarrow \arg\max_{i \in \hat{N}, v \in V} \{\zeta_{iv}\}$
    if $\tau_{v^*} + t_{r_{v^*}, i^*} + t_{i^*, (n+1)} \leq T_{\max}$ **then**
        $s_{i^*} = \min(s_i, T_{\max} - (\tau_{v^*} + t_{r_{v^*}, i^*} + t_{i^*, (n+1)}))$
        Update $\tau_{v^*} \leftarrow \tau_{v^*} + t_{r_{v^*}, i^*} + s_{i^*}$, $\quad r_{v^*} \leftarrow i^*$
        $R^{v^*} = \{0, \dots, i^*, n+1\}$, $S^{v^*} = \{0, \dots, s_{i^*}, 0\}$, $\quad \hat{N} \leftarrow \hat{N} \setminus i^*$
    else
        $\hat{V} \leftarrow \hat{V} \setminus v^*$
    end if
end while
Output: *sol*

A.2 Perturbation phase

The perturbation method is a key component in ILS where a perturbation is carried out on the local optimal solution. The main idea is to remove some visited customers and then insert some unvisited customers to generate a diversified solution.

We define a perturbation strength parameter π to indicate the number of customers to be removed from the routes. The range of π belongs to $[\pi_{\min}, \pi_{\max}] = [|V|/2, |N(sol)|/2]$. Initially, we set $\pi = \pi_{\min}$ and increase its value by 1 if no better solution is found. Once a new best solution is found, π is reset to $\pi_{\min}$. This way, the search focuses on the promising region around the local optima during the early stage while it moves to the unexplored region further away from the local optima at a later stage. Once the number of customers to be removed is determined, we then randomly select π customers and remove them from the local optimal solution.

To diversify the search to unexplored region, we insert some unvisited customers into the current solution after removal. First, to define an unexplored region, we assign a number to each customer to record the times it has been inserted into the local optimal solution successfully in the past perturbation phase. Then we sort all unvisited customers in ascending order of this number, and the customers at the top of the list are more likely to be in the unexplored region. Therefore, the algorithm proceeds to check these customers successively. For each customer, it attempts to find all inserting positions satisfying all constraints. Note that the amount of service time of this inserting customer is supposed to be zero. Then, it randomly chooses a feasible position to insert this customer. If no such position can be found, then it turns to the next customer. The insertion will continue until all customers in the unexplored region have been checked or the total travel time of the current route beyond half of $T_{\max}$.

A.3 Local search phase

Local search is another important part in ILS, here we define eight neighbourhood structures based on the classical neighborhoods in routing problem and the characteristic of our problem. A given solution can be improved by adjusting the order of customers in a route to reduce the travel time, by replacing the position of a customer in order to reduce the delay penalty, or by removing some customers increase the service time spent at other customers. We choose some classical operators for the routing problem and also design additional neighborhood search operators. They are presented as follows:

- *Or-opt*: Relocate a chain of σ consecutive customers in the same route, we set $\sigma = 1, 2, 3$ here.
- *swap*: Swap the position of two customers in the same route.
- *2-opt*: Remove two non-adjacent edges from the route and replace with other two edges, we notice that the sequence between these two removed edges are reversed.
- λ-*Exchange*: Change the locations of two chains of λ consecutive customers located in a pair of routes (r_i, r_j), we set $\lambda \leq \lceil \min\{|r_i|, |r_j|\}/2 \rceil$.
- *Cross*: Remove two edges from two routes and replace them with other two edges.
- *Removal*: Remove all visited but unserved customers from the solution to extend the service time.
- *Insertion*: Insert an unvisited customer into the solution to increase the profit.
- *Replacement*: Replace a visited customer v_i by an unvisited customer v_j with the possibility $\min\{1, \frac{p_j}{p_i}\}$

The first three neighborhood search operators are intra-route operators. These operators aim to improve the profit of a route by adjusting the order of visiting customers. The following two are inter-route operators. They are used to balance the whole profit of a solution via changing customers to different routes. While the last three operators are designed to change the set of visited customers. The removal operator will remove unserved customers from every route in the solution since we know that traversing a customer without any service is time-wasting since no return will be obtained. Thus, the solution will be improved if there exist an unserved customer. After removing unserved customers, the solution may have a place to accommodate other unvisited customers, thus insertion is called following the removal operator. We also attempt to improve the current solution by replacing visited customers by unvisited customers. In each iteration, we implement these eight neighborhoods in the given sequence, and for every neighborhood, we use the best improvement strategy and return the best improved neighborhood if any.

A.3.1 Rules to reduce the neighbourhood size.

In a neighbourhood operator, many neighbours of the current solution or route are generated. Consider the complete evaluation of a solution is time-consuming, we set two rules to discard some uninteresting neighbours as follows. Let sol' denotes a neighbour generated from solution sol using the move operator.

Rule 1: Given the solution sol, in which the service times of all customers are known, the total collected profit $f(sol)$ can be calculated exactly using the objective function. If we assume that the profit at each customer can be immediately obtained when the vehicle arrives at the customer without spending any amount of service time. Then an upper bound of the total profit $f_{UB}(sol')$ of the neighbour sol' can be calculated as $\sum_{v_i \in N_{sol'}} (p_{v_i} - d_{v_i} a_{v_i})$, where $N_{sol'}$ is the set of all visited customers by solution sol' and a_{v_i} is the arrival time at visited customer v_i without any service time. Then we define the neighbour solution sol' is an uninteresting neighbour if $f_{UB}(sol') \leq f(sol)$, as it is obvious that this neighbour solution cannot result in a higher profit than the current one.

Rule 2: As the service time take a crucial role in collecting the profit, we consider sol' as an uninteresting neighbour if the amount of total available service time $\hat{T}'$ left in the neighbour solution sol' is less than 50% of the amount $\hat{T}$ in the current solution sol.

B Details of the instance

We generate the instances based on the Solomon's instances (Available from http://neo.lcc.uma.es/vrp/vrp-instances/capacitated-vrp-with-time-windows-instances/). Specifically, we consider the R(C) type network extracting from R101(C101) data file with first 25 customers. In each instance, the parameters, i.e., x coordinate x_i, y coordinate y_i and demand p_i (the ideal profit in our problem) are taken from Solomon's instances. We take the first node to be the starting and ending depot, and we compute t_{ij} using the Euclidean distance and round it to the nearest hundredth. The other parameters in our problem are conducted from the location (x_i, y_i) using a similar generation scheme as in Erdoğan and Laporte (2013). More specifically, let $X_{\max}, X_{\min}, Y_{\max}$ and $Y_{\min}$ denote the maximum X coordinate, minimum X coordinate, maximum Y coordinate and minimum Y coordinate of all customers in the instance, respectively. Then we determine a time factor as $L = 2\lfloor X_{\max} - X_{\min} + Y_{\max} - Y_{\min}\rfloor$. Next we set two factors as $r_{\min} = 0.2$ and $r_{\max} = 0.5$ and determine the delay rate as $d_i = (r_{\min} + ((\lfloor X_i + Y_i\rfloor \bmod 30)/30) * (r_{\max} - r_{\min}))/25$. The collecting coefficient β_i is also determined in the similar manner, using the formula $\beta_i = (r_{\min} + ((\lfloor X_i + Y_i\rfloor \bmod 20)/20) * (r_{\max} - r_{\min}))/20$.

For each type of network, we generate a total of 48 instances. Each instance is characterized by the number of customers n, the number of vehicles V and the length of decision horizon $T_{\max}$: The number of customers n belongs to $\{6, 9, 25\}$. The number of vehicles V belongs to $\{2, 3, 4, 5\}$ and the length of time limit equal to $T_{\max} \in \{L, 2L, 3L, 4L\}$. Here we notice that the 6(9) customers are the first 6(9) among 25 customers in R type network, while for the C type network, the 6(9) customers are chosen from 2(3) clusters.

C Detailed computational results

The detailed results on the exact method BBC and its enhancement are presented in Section C.1. Section C.2 describes the route generation for the service time scheduling problem and the parameters using in the different NLP methods. Section C.3 provides the detailed results on the ILS-MCS.

C.1 Detailed results on the exact methods BBC and enhanced BBC

Tables 6–8 present the performances of the SCIP, the BBC and the enhanced BBC for the instances with 6, 9 and 25 customers. We highlight the best found objective value of each instances using bold font in Tables 7 and 8.

C.2 Experimental settings for the comparisons of the algorithms for the service time scheduling subproblem

We demonstrate the effectiveness of the MCS by comparing it with other NLP methods on a predetermined group of routes. The number of customers in a route varies from 2 to 30, and five routes are conducted randomly for each number of customers. We use C101 data file which includes 100 customers and 1 depot to generate these routes and define $T_{\max} = 4L$ for all routes, where L is perimeter of the minimum bounding rectangle of the 101 nodes.

We performed preliminary experiments for hyper-parameter tuning for all the algorithms to determine the best-performing ones. The following parameters are used in the MCS: $\alpha = 10$, $\alpha_{\min} = 1$ and $||\vec{e_i}|| = 0.01$. For the IpOpt solver, we use the default settings (as it generally gives the best results) where the IpOpt solver terminates when the NLP optimality error becomes smaller than 10^{-3}. For the MMA, SLSQP, COBYLA and ISRES methods available from Johnson (2019), we set the relative tolerance on function value as 10^{-7} and the maximum wall-clock time as 1 second, so the algorithm stops before 1 second if the increase in the function value from one iteration to next relative to the function value is less than 10^{-7}. For the NOMAD and CS methods available in Abramson et al. (2019), we set the maximum wall-clock time as 1 second, the direction type as *OrthoMADS_2N* for the NOMAD method and *GPS_2N_STATIC* for the CS method, we refer readers to Audet et al. (2009) for more details about these parameters.

Table 6: Results comparing three exact methods on instances with 6 customers

Instance	SCIP		BBC		Enhanced BBC	
	z^*	Time(s)	z^*	Time(s)	z^*	Time(s)
C/6/70/2	15.88	34.2	15.88	25.9	15.88	0.4
C/6/140/2	30.25	79.1	30.25	47.7	30.25	2.0
C/6/210/2	40.90	225.4	40.90	60.8	40.90	4.7
C/6/280/2	48.01	453.5	48.01	53.8	48.01	10.2
C/6/70/3	20.66	25.6	20.66	13.0	20.66	0.9
C/6/140/3	40.91	50.5	40.91	18.4	40.91	1.0
C/6/210/3	52.74	68.5	52.74	22.8	52.74	3.4
C/6/280/3	59.52	89.6	59.52	20.0	59.52	2.2
C/6/70/4	25.25	11.1	25.25	6.5	25.25	0.5
C/6/140/4	48.52	10.1	48.52	6.9	48.52	0.9
C/6/210/4	60.83	17.7	60.83	5.1	60.83	1.1
C/6/280/4	66.82	18.2	66.82	4.9	66.82	1.1
C/6/70/5	29.28	5.8	29.28	1.4	29.28	0.2
C/6/140/5	54.99	2.8	54.99	1.4	54.99	0.3
C/6/210/5	66.57	2.7	66.57	1.9	66.57	0.3
C/6/280/5	71.71	1.8	71.71	1.6	71.71	0.4
Avg	*45.80*	*68.5*	*45.80*	*18.3*	*45.80*	*1.9*
R/6/144/2	35.67	51.0	35.67	28.4	35.67	0.3
R/6/288/2	52.98	121.2	52.98	52.1	52.98	3.4
R/6/432/2	60.98	173.2	60.98	60.2	60.98	7.8
R/6/576/2	63.70	155.6	63.70	53.9	63.70	14.1
R/6/144/3	45.58	43.1	45.58	19.1	45.58	0.5
R/6/288/3	63.83	42.5	63.83	20.3	63.83	2.0
R/6/432/3	68.66	50.8	68.66	24.9	68.66	4.0
R/6/576/3	69.85	47.2	69.85	21.4	69.85	5.5
R/6/144/4	52.21	14.5	52.21	4.6	52.21	0.6
R/6/288/4	69.09	10.0	69.09	4.7	69.09	1.2
R/6/432/4	72.29	12.7	72.29	4.5	72.29	1.9
R/6/576/4	72.85	15.4	72.85	4.9	72.85	2.7
R/6/144/5	58.05	4.6	58.05	1.1	58.05	0.5
R/6/288/5	72.50	4.9	72.50	1.0	72.50	0.7
R/6/432/5	74.72	4.7	74.72	1.0	74.72	0.8
R/6/576/5	75.08	4.5	75.08	1.2	75.08	1.2
Avg	*63.00*	*47.2*	*63.00*	*19.0*	*63.00*	*2.9*

Table 7: Results comparing three exact methods on instances with 9 customers

Instance	SCIP				BBC				Enhanced BBC			
	z_{lb}	z_{ub}	Gap(%)	Time(s)	z_{lb}	z_{ub}	Gap(%)	Time(s)	z_{lb}	z_{ub}	Gap(%)	Time(s)
C/9/116/2	35.69	87.03	143.9	7200.0	**36.32**	88.69	144.2	7200.0	**36.32**	36.32	0.0	2.9
C/9/232/2	62.29	128.27	105.9	7200.0	62.98	128.27	103.7	7200.0	**63.68**	63.69	0.0	196.2
C/9/348/2	80.32	133.73	66.5	7200.0	80.25	135.09	68.3	7200.0	**80.57**	85.60	6.2	7200.0
C/9/464/2	89.48	135.92	51.9	7200.0	90.35	136.54	51.1	7200.0	**90.53**	109.26	20.7	7200.0
C/9/116/3	46.28	86.98	87.9	7200.0	45.73	88.69	94.0	7200.0	**46.47**	46.47	0.0	6.9
C/9/232/3	77.69	128.27	65.1	7200.0	80.73	128.27	58.9	7200.0	**81.46**	81.47	0.0	862.7
C/9/348/3	96.00	135.09	40.7	7200.0	98.34	135.09	37.4	7200.0	**98.78**	101.73	3.0	7200.0
C/9/464/3	106.48	136.54	28.2	7200.0	106.63	136.54	28.0	7200.0	**106.82**	127.48	19.3	7200.0
C/9/116/4	**56.44**	56.44	0.0	5990.7	56.40	88.65	57.2	7200.0	**56.44**	56.44	0.0	15.1
C/9/232/4	94.26	126.24	33.9	7200.0	95.06	128.27	34.9	7200.0	**95.26**	95.27	0.0	565.9
C/9/348/4	110.36	134.16	21.6	7200.0	110.70	135.09	22.0	7200.0	**110.96**	110.97	0.0	1647.5
C/9/464/4	106.23	136.54	28.5	7200.0	116.85	136.54	16.9	7200.0	**116.98**	116.99	0.0	6840.9
C/9/116/5	**66.17**	66.17	0.0	2138.5	**66.17**	66.18	0.0	3807.7	**66.17**	66.18	0.0	24.0
C/9/232/5	**105.12**	121.45	15.5	7200.0	**105.21**	105.21	0.0	4792.3	**105.21**	105.22	0.0	381.0
C/9/348/5	117.78	134.17	13.9	7200.0	**118.94**	118.95	0.0	4600.5	**118.94**	118.95	0.0	1117.9
C/9/464/5	117.25	136.54	16.5	7200.0	**123.00**	123.01	0.0	3882.4	**123.00**	123.01	0.0	1601.0
Avg	*85.49*	*117.72*	*45.0*	*6808.08*	*87.10*	*117.44*	*44.8*	*6467.7*	*87.35*	*90.31*	*3.1*	*2628.9*
R/9/176/2	40.23	87.53	117.6	7200.0	39.72	92.54	133.0	7200.0	**40.39**	40.39	0.0	5.2
R/9/352/2	61.99	102.94	66.0	7200.0	64.28	104.10	62.0	7200.0	**64.92**	64.92	0.0	2451.3
R/9/528/2	70.28	105.05	49.5	7200.0	72.99	105.05	43.9	7200.0	**73.99**	87.73	18.6	7200.2
R/9/704/2	76.13	105.00	37.9	7200.0	75.80	105.17	38.7	7200.0	**76.71**	93.19	21.5	7200.1
R/9/176/3	**56.12**	90.50	61.3	7200.0	55.63	92.54	66.3	7200.0	**56.12**	56.12	0.0	12.6
R/9/352/3	79.23	103.32	30.4	7200.0	**80.09**	104.10	30.0	7200.0	**80.09**	80.10	0.0	3517.1
R/9/528/3	81.14	105.05	29.5	7200.0	84.31	105.05	24.6	7200.0	**86.45**	93.63	8.3	7200.0
R/9/704/3	85.83	105.17	22.5	7200.0	**87.47**	105.17	20.2	7200.0	**87.47**	98.33	12.4	7200.4
R/9/176/4	**65.99**	65.99	0.0	2682.7	65.62	92.54	41.0	7200.0	**65.99**	65.99	0.0	166.7
R/9/352/4	**88.46**	102.66	16.1	7200.0	88.12	104.10	18.1	7200.0	**88.46**	88.47	0.0	1797.8
R/9/528/4	91.69	104.76	14.3	7200.0	91.72	105.05	14.5	7200.0	**92.57**	92.58	0.0	2832.4
R/9/704/4	92.19	105.17	14.1	7200.0	**93.20**	105.17	12.8	7200.0	**93.20**	93.21	0.0	5808.0
R/9/176/5	**74.14**	74.14	0.0	3377.5	**74.14**	74.15	0.0	4677.4	**74.14**	74.15	0.0	74.8
R/9/352/5	88.46	102.66	16.1	7200.0	**93.92**	93.93	0.0	4224.3	**93.92**	93.93	0.0	560.3
R/9/528/5	**96.29**	102.67	6.6	7200.0	**96.29**	96.30	0.0	3923.0	**96.29**	96.30	0.0	895.5
R/9/704/5	96.39	105.15	9.1	7200.0	**96.55**	96.56	0.0	3972.1	**96.55**	96.56	0.0	1243.6
Avg	*77.79*	*97.99*	*30.7*	*6678.76*	*78.74*	*98.84*	*31.6*	*6449.8*	*79.20*	*82.23*	*3.8*	*3010.4*

Table 8: Results comparing three exact methods on instances with 25 customers

Instance	SCIP				BBC				Enhanced BBC			
	z_{lb}	z_{ub}	Gap(%)	Time(s)	z_{lb}	z_{ub}	Gap(%)	Time(s)	z_{lb}	z_{ub}	Gap(%)	Time(s)
C/25/130/2	60.92	317.85	421.71	7200.0	**62.03**	317.85	412.4	7200.0	59.34	74.02	24.7	7200.0
C/25/260/2	95.72	430.34	349.59	7200.0	102.17	430.33	321.2	7200.0	**122.40**	145.37	18.8	7200.0
C/25/390/2	153.47	447.92	191.87	7200.0	120.11	447.92	272.9	7200.0	**168.38**	224.03	33.0	7200.0
C/25/520/2	180.64	451.31	149.84	7200.0	127.45	451.31	254.1	7200.0	**199.68**	236.35	18.4	7200.0
C/25/130/3	71.12	317.85	346.91	7200.0	**87.43**	317.85	263.5	7200.0	85.33	109.74	28.6	7200.0
C/25/260/3	134.39	430.33	220.20	7200.0	137.95	430.33	211.9	7200.0	**167.26**	195.53	16.9	7200.0
C/25/390/3	189.96	447.92	135.80	7200.0	159.39	447.92	181.0	7200.0	**215.61**	252.67	17.2	7200.0
C/25/520/3	227.10	451.30	98.72	7200.0	167.63	451.31	169.2	7200.0	**250.22**	296.36	18.4	7200.0
C/25/130/4	87.29	317.85	264.14	7200.0	**102.97**	317.85	208.7	7200.0	102.68	139.19	35.6	7200.0
C/25/260/4	167.85	430.34	156.38	7200.0	170.11	430.33	153.0	7200.0	**192.78**	234.26	21.5	7200.0
C/25/390/4	213.86	447.92	109.44	7200.0	196.64	447.92	127.8	7200.0	**246.15**	366.80	49.0	7200.0
C/25/520/4	255.27	451.32	76.80	7200.0	206.55	451.31	118.5	7200.0	**270.04**	429.75	59.1	7200.0
C/25/130/5	97.86	317.85	224.80	7200.0	115.52	317.85	175.2	7200.0	**119.61**	162.42	35.8	7200.0
C/25/260/5	189.25	430.33	127.39	7200.0	192.01	430.33	124.1	7200.0	**224.62**	266.59	18.7	7200.0
C/25/390/5	245.91	447.92	82.15	7200.0	221.29	447.92	102.4	7200.0	**271.47**	422.02	55.5	7200.0
C/25/520/5	283.68	451.30	59.09	7200.0	232.13	451.31	94.4	7200.0	**303.75**	451.31	48.6	7200.0
Avg	*165.89*	*411.85*	*188.43*	*7200.00*	*150.08*	*411.85*	*199.4*	*7200.0*	*187.46*	*250.40*	*31.2*	*7200.0*
R/25/240/2	32.59	303.60	831.47	7200.0	64.72	303.60	369.1	7200.0	**77.12**	79.59	3.2	7200.0
R/25/480/2	82.53	322.40	290.67	7200.0	90.09	322.40	257.9	7200.0	**124.86**	150.88	20.8	7200.0
R/25/720/2	104.75	323.41	208.74	7200.0	94.14	323.41	243.5	7200.0	**145.95**	199.02	36.4	7200.0
R/25/960/2	94.48	323.48	242.39	7200.0	94.66	323.48	241.7	7200.0	**158.29**	234.43	48.1	7200.0
R/25/240/3	73.00	303.61	315.90	7200.0	91.54	303.60	231.6	7200.0	**103.68**	115.36	11.3	7200.0
R/25/480/3	131.31	322.40	145.53	7200.0	141.80	322.40	127.4	7200.0	**164.02**	197.97	20.7	7200.0
R/25/720/3	145.26	323.41	122.64	7200.0	154.08	323.41	109.9	7200.0	**170.60**	298.91	75.2	7200.0
R/25/960/3	144.33	323.49	124.13	7200.0	156.86	323.48	106.2	7200.0	**172.43**	323.48	87.6	7200.0
R/25/240/4	81.00	303.60	274.82	7200.0	120.92	303.60	151.1	7200.0	**126.02**	150.89	19.7	7200.0
R/25/480/4	161.19	322.39	100.01	7200.0	**177.98**	322.40	81.1	7200.0	175.47	297.17	69.4	7200.0
R/25/720/4	188.53	323.40	71.54	7200.0	191.83	323.41	68.6	7200.0	**201.27**	323.41	60.7	7200.0
R/25/960/4	98.41	323.48	228.70	7200.0	**194.70**	323.48	66.1	7200.0	193.57	323.48	67.1	7200.0
R/25/240/5	102.51	303.60	196.16	7200.0	139.49	303.60	117.6	7200.0	**143.08**	214.78	50.1	7200.0
R/25/480/5	157.68	322.40	104.47	7200.0	**205.82**	322.40	56.6	7200.0	197.44	319.40	61.8	7200.0
R/25/720/5	194.04	323.40	66.67	7200.0	**221.34**	323.41	46.1	7200.0	219.05	323.41	47.6	7200.0
R/25/960/5	-	-	-	7200.0	**224.30**	323.48	44.2	7200.0	224.28	323.48	44.2	7200.0
Avg	*119.44*	*317.87*	*221.59*	*7200.00*	*142.66*	*317.87*	*151.6*	*7200.0*	*158.19*	*236.81*	*45.3*	*7200.0*

- indicates that SCIP did not find a solution within 2 hours.

C.3 Detailed results of the heuristic method ILS-MCS

C.3.1 Detailed results of ILS-MCS and ILS-IpOpt on small instances (6, 9 and 25 Customers)

The results of heuristic methods ILS-MCS and ILS-IpOpt on Instances with 6, 9 and 25 customers are shown in Tables 9–11, we also list the results of the enhanced BBC in these three tables for easy comparison.

Table 9: Results of the heuristic methods on the instance with 6 Customers

Instance	Enhanced BBC z_{lb}	Enhanced BBC Gap (%)	Enhanced BBC Found Time(s)	ILS-MCS Best Obj	ILS-MCS Worst Obj	ILS-MCS Avg. Obj	ILS-MCS Std. Dev.	ILS-MCS Avg. Time(s)	ILS-IpOpt Best Obj	ILS-IpOpt Worst Obj	ILS-IpOpt Avg. Obj	ILS-IpOpt Std. Dev.	ILS-IpOpt Avg. Time(s)
C6/70/2	**15.88**	0.0	0.36	**15.88**	15.88	15.88	0.00	0.03	**15.88**	15.88	15.88	0.00	0.98
C6/140/2	**30.25**	0.0	1.77	**30.25**	30.25	30.25	0.00	0.05	**30.25**	30.25	30.25	0.00	3.07
C6/210/2	**40.90**	0.0	2.47	**40.90**	40.90	40.90	0.00	0.06	**40.90**	40.90	40.90	0.00	2.94
C6/280/2	**48.01**	0.0	1.43	**48.01**	48.01	48.01	0.00	0.09	**48.01**	48.01	48.01	0.00	3.99
C6/70/3	**20.66**	0.0	0.85	**20.65**	20.65	20.65	0.00	0.05	**20.65**	20.65	20.65	0.00	1.67
C6/140/3	**40.91**	0.0	0.36	**40.91**	40.91	40.91	0.00	0.06	**40.91**	40.91	40.91	0.00	1.67
C6/210/3	**52.74**	0.0	2.85	**52.74**	52.74	52.74	0.00	0.06	**52.74**	52.74	52.74	0.00	1.71
C6/280/3	**59.52**	0.0	0.79	**59.52**	59.52	59.52	0.00	0.06	**59.52**	59.52	59.52	0.00	1.78
C6/70/4	**25.25**	0.0	0.44	**25.25**	25.25	25.25	0.00	0.04	**25.25**	25.25	25.25	0.00	0.60
C6/140/4	**48.52**	0.0	0.72	**48.52**	48.52	48.52	0.00	0.08	**48.52**	48.52	48.52	0.00	1.01
C6/210/4	**60.83**	0.0	0.94	**60.83**	60.83	60.83	0.00	0.06	**60.83**	60.83	60.83	0.00	1.00
C6/280/4	**66.82**	0.0	0.81	**66.82**	66.82	66.82	0.00	0.05	**66.82**	66.75	66.79	0.03	0.97
C6/70/5	**29.28**	0.0	0.13	**29.28**	29.28	29.28	0.00	0.03	**29.28**	29.28	29.28	0.00	0.54
C6/140/5	**54.99**	0.0	0.28	**54.93**	54.93	54.93	0.00	0.03	**54.93**	54.93	54.93	0.00	0.45
C6/210/5	**66.57**	0.0	0.13	**66.57**	66.57	66.57	0.00	0.04	**66.57**	66.57	66.57	0.00	0.52
C6/280/5	**71.71**	0.0	0.07	**71.71**	71.71	71.71	0.00	0.04	**71.71**	71.71	71.71	0.00	0.48
Avg	*45.80*	*0.0*	*0.90*	*45.80*	*45.80*	*45.80*	*0.00*	*0.05*	*45.80*	*45.79*	*45.80*	*0.00*	*1.46*
R6/144/2	**35.67**	0.0	0.18	**35.67**	35.67	35.67	0.00	0.10	**35.67**	35.67	35.67	0.00	1.76
R6/288/2	**52.98**	0.0	1.61	**52.98**	52.98	52.98	0.00	0.08	**52.98**	52.98	52.98	0.00	4.72
R6/432/2	**60.98**	0.0	0.44	**60.98**	60.98	60.98	0.00	0.09	**60.98**	60.98	60.98	0.00	5.08
R6/576/2	**63.70**	0.0	1.83	**63.70**	63.70	63.70	0.00	0.14	**63.70**	63.70	63.70	0.00	3.90
R6/144/3	**45.58**	0.0	0.38	**45.58**	45.58	45.58	0.00	0.08	**45.58**	45.58	45.58	0.00	1.82
R6/288/3	**63.83**	0.0	1.22	**63.83**	63.83	63.83	0.00	0.08	**63.83**	63.83	63.83	0.00	2.02
R6/432/3	**68.66**	0.0	3.44	**68.66**	68.66	68.66	0.00	0.06	**68.66**	68.66	68.66	0.00	1.90
R6/576/3	**69.85**	0.0	4.78	**69.85**	69.85	69.85	0.00	0.07	**69.85**	69.85	69.85	0.00	2.16
R6/144/4	**52.21**	0.0	0.48	**52.21**	52.21	52.21	0.00	0.04	**52.21**	52.16	52.20	0.02	0.78
R6/288/4	**69.09**	0.0	0.74	**69.09**	69.09	69.09	0.00	0.04	**69.09**	69.03	69.08	0.02	1.04
R6/432/4	**72.29**	0.0	1.43	**72.29**	72.15	72.10	0.08	0.05	**72.29**	71.68	72.13	0.24	1.10
R6/576/4	**72.85**	0.0	2.66	**72.85**	72.77	72.53	0.14	0.06	**72.85**	72.18	72.54	0.22	1.24
R6/144/5	**58.05**	0.0	0.52	**58.05**	58.05	58.05	0.00	0.04	**58.05**	58.05	58.05	0.00	0.56
R6/288/5	**72.50**	0.0	0.43	**72.50**	72.50	72.50	0.00	0.05	**72.50**	72.50	72.50	0.00	0.62
R6/432/5	**74.72**	0.0	0.64	**74.72**	74.72	74.72	0.00	0.04	**74.72**	74.30	74.55	0.21	0.62
R6/576/5	**75.08**	0.0	0.72	**75.08**	75.08	75.08	0.00	0.07	**75.08**	75.08	75.08	0.00	0.73
Avg	*63.00*	*0.0*	*1.34*	*63.00*	*62.99*	*62.97*	*0.01*	*0.07*	*63.00*	*62.89*	*62.96*	*0.04*	*1.88*

Table 10: Results of the heuristic methods on the instance with 9 customers

	Enhanced BBC			ILS-MCS					ILS-IpOpt				
Instance	z_{lb}	Gap (%)	Found Time(s)	Best Obj	Worst Obj	Avg. Obj	Std. Dev.	Avg. Time(s)	Best Obj	Worst Obj	Avg. Obj	Std. Dev.	Avg. Time(s)
C9/116/2	**36.32**	0.0	2.85	**36.32**	36.32	36.32	0.00	0.05	**36.32**	36.32	36.32	0.00	3.94
C9/232/2	**63.68**	0.0	5.28	**63.68**	63.13	63.54	0.24	0.25	**63.68**	63.64	63.67	0.02	23.33
C9/348/2	**80.57**	6.2	698.65	**80.57**	80.57	80.57	0.00	0.54	**80.57**	80.57	80.57	0.00	84.87
C9/464/2	**90.53**	20.7	148.67	**90.53**	90.53	90.53	0.00	0.44	**90.53**	90.53	90.53	0.00	57.13
C9/116/3	**46.47**	0.0	6.11	**46.47**	46.47	46.47	0.00	0.14	**46.47**	46.47	46.47	0.00	6.96
C9/232/3	**81.46**	0.0	522.02	**81.46**	81.46	81.46	0.00	0.30	**81.46**	81.46	81.46	0.00	9.92
C9/348/3	**98.78**	3.0	6381.63	**98.78**	98.78	98.78	0.00	0.34	**98.78**	98.76	98.78	0.01	15.01
C9/464/3	**106.82**	19.3	598.13	**106.86**	106.86	106.86	0.00	0.52	**106.86**	106.86	106.86	0.00	20.45
C9/116/4	**56.44**	0.0	11.98	**56.44**	56.06	56.02	0.12	0.32	**56.44**	55.58	56.12	0.29	5.21
C9/232/4	**95.26**	0.0	519.61	**95.26**	95.26	95.26	0.00	0.26	**95.26**	95.26	95.26	0.00	8.37
C9/348/4	**110.96**	0.0	1474.63	**110.96**	110.96	110.96	0.00	0.28	**110.96**	110.96	110.96	0.00	9.60
C9/464/4	**116.98**	0.0	2345.14	**116.98**	116.98	116.98	0.00	0.49	**116.98**	116.98	116.98	0.00	11.31
C9/116/5	**66.17**	0.0	14.80	**66.17**	66.17	66.17	0.00	0.12	**66.17**	65.75	66.09	0.17	3.62
C9/232/5	**105.21**	0.0	77.77	**105.20**	105.20	105.20	0.00	0.15	**105.20**	105.12	105.17	0.04	3.83
C9/348/5	**118.94**	0.0	216.98	**118.94**	118.94	118.94	0.00	0.18	**118.94**	118.71	118.89	0.09	4.20
C9/464/5	**123.00**	0.0	1185.06	**123.00**	123.00	123.00	0.00	0.21	**123.00**	122.51	122.82	0.18	4.63
Avg	*87.35*	*3.1*	*888.08*	*87.35*	*87.29*	*87.32*	*0.02*	*0.29*	*87.35*	*87.22*	*87.31*	*0.05*	*17.03*
R9/176/2	**40.39**	0.0	3.58	**40.39**	40.39	40.39	0.00	0.04	**40.39**	40.39	40.39	0.00	4.85
R9/352/2	**64.92**	0.0	164.35	**64.92**	64.92	64.92	0.00	0.18	**64.92**	64.92	64.92	0.00	14.65
R9/528/2	**73.99**	18.6	101.08	**73.99**	73.99	73.99	0.00	0.40	**73.99**	73.49	73.89	0.20	21.10
R9/704/2	76.71	21.5	1165.03	**76.93**	76.71	76.88	0.10	0.51	**76.93**	76.67	76.88	0.10	31.38
R9/176/3	**56.12**	0.0	4.75	**56.12**	56.12	56.12	0.00	0.13	**56.12**	55.95	56.05	0.08	5.80
R9/352/3	**80.09**	0.0	1049.94	**80.09**	80.09	80.09	0.00	0.19	**80.09**	80.09	80.09	0.00	9.88
R9/528/3	**86.45**	8.3	909.40	**86.45**	86.45	86.45	0.00	0.17	**86.45**	86.25	86.41	0.08	12.59
R9/704/3	**87.47**	12.4	2578.38	**87.47**	87.47	87.47	0.00	0.39	**87.47**	87.47	87.47	0.00	13.80
R9/176/4	**65.99**	0.0	84.84	**65.99**	65.99	65.99	0.00	0.08	**65.99**	65.99	65.99	0.00	5.54
R9/352/4	**88.46**	0.0	203.20	**88.46**	88.46	88.46	0.00	0.14	**88.46**	88.46	88.46	0.00	6.25
R9/528/4	**92.57**	0.0	454.31	**92.57**	92.57	92.57	0.00	0.17	**92.57**	92.57	92.57	0.00	8.44
R9/704/4	**93.20**	0.0	1351.38	**93.20**	93.10	93.18	0.04	0.37	**93.20**	93.20	93.20	0.00	9.96
R9/176/5	**74.14**	0.0	50.44	**74.14**	74.14	74.14	0.00	0.15	**74.14**	73.79	73.95	0.16	3.84
R9/352/5	**93.92**	0.0	96.85	**93.92**	93.92	93.92	0.00	0.18	**93.92**	92.76	93.58	0.46	4.79
R9/528/5	**96.29**	0.0	813.91	**96.29**	96.05	96.23	0.10	0.22	96.14	95.96	96.04	0.08	4.91
R9/704/5	**96.55**	0.0	729.64	**96.55**	96.21	96.47	0.15	0.20	95.92	95.42	95.73	0.19	5.45
Avg	*79.20*	*3.8*	*610.07*	*79.22*	*79.16*	*79.20*	*0.02*	*0.22*	*79.17*	*78.96*	*79.10*	*0.09*	*10.20*

Table 11: Results of the heuristic methods on instance with 25 customers

Instance	Enhanced BBC			ILS-MCS					ILS-IpOpt				
	z_{lb}	Gap (%)	Found Time(s)	Best Obj	Worst Obj	Avg. Obj	Std. Dev.	Avg. Time(s)	Best Obj	Worst Obj	Avg. Obj	Std. Dev.	Avg. Time(s)
C25/130/2	59.34	24.7	3564.46	**65.76**	65.68	65.71	0.03	0.43	**65.76**	65.76	65.76	0.00	15.75
C25/260/2	122.40	18.8	5446.90	**129.02**	129.02	129.02	0.00	1.66	**129.02**	129.02	129.02	0.00	86.95
C25/390/2	168.38	33.0	4652.80	173.89	173.56	173.74	0.15	2.12	**173.93**	173.20	173.78	0.29	411.95
C25/520/2	199.68	18.4	453.42	205.45	205.45	205.45	0.00	8.60	**205.61**	205.47	205.55	0.07	474.28
C25/130/3	85.33	28.6	3211.66	**90.46**	90.43	90.39	0.03	0.85	**90.46**	90.46	90.46	0.00	37.31
C25/260/3	167.26	16.9	5301.15	171.10	170.98	171.01	0.05	2.20	**171.12**	171.12	171.12	0.00	266.46
C25/390/3	215.61	17.2	6877.86	221.60	221.42	221.50	0.08	3.66	**221.67**	221.43	221.57	0.12	732.88
C25/520/3	250.22	18.4	2.29	253.74	253.70	253.73	0.02	41.64	**253.94**	253.45	253.80	0.18	785.75
C25/130/4	102.68	35.6	5740.65	**110.72**	110.64	110.69	0.03	0.62	**110.72**	110.72	110.72	0.00	36.86
C25/260/4	192.78	21.5	1883.23	**206.37**	206.29	206.25	0.06	3.35	**206.37**	206.32	206.35	0.02	222.62
C25/390/4	246.15	49.0	807.27	258.74	258.59	258.68	0.05	9.94	**258.83**	258.58	258.74	0.08	541.21
C25/520/4	270.04	59.1	3379.70	**290.24**	289.85	290.13	0.16	20.36	290.21	289.01	289.52	0.45	495.86
C25/130/5	119.61	35.8	6073.63	**129.39**	129.31	129.36	0.03	1.03	**129.39**	129.39	129.39	0.00	50.34
C25/260/5	224.62	18.7	5440.23	**233.75**	233.73	233.74	0.01	4.06	**233.75**	233.58	233.67	0.07	293.30
C25/390/5	271.47	55.5	292.13	**287.98**	287.93	287.97	0.02	12.05	287.93	287.62	287.76	0.11	555.85
C25/520/5	303.75	48.6	6410.86	**318.35**	318.18	318.29	0.07	23.64	318.16	317.42	317.88	0.26	454.29
Avg	*187.46*	*31.2*	*3721.14*	*196.66*	*196.55*	*196.60*	*0.05*	*8.51*	*196.68*	*196.41*	*196.57*	*0.10*	*341.35*
R25/240/2	77.12	3.2	292.13	**77.12**	77.12	77.12	0.00	0.50	**77.12**	77.12	77.12	0.00	22.28
R25/480/2	124.86	20.8	6410.86	**128.05**	125.76	126.33	0.99	1.95	**128.05**	125.76	127.59	0.92	160.66
R25/720/2	145.95	36.4	6261.31	**155.73**	155.65	155.71	0.03	11.68	**155.73**	153.97	154.92	0.68	308.92
R25/960/2	158.29	48.1	2973.17	**169.58**	169.28	169.48	0.12	52.42	**169.58**	169.28	169.47	0.11	704.35
R25/240/3	103.68	11.3	5068.15	**104.50**	104.49	104.50	0.00	0.98	**104.50**	104.50	104.50	0.00	51.58
R25/480/3	164.02	20.7	2922.62	**168.56**	167.65	168.12	0.45	6.28	**168.56**	167.79	168.36	0.29	311.35
R25/720/3	170.60	75.2	5438.71	**195.07**	195.07	195.07	0.00	40.34	**195.07**	194.66	194.88	0.16	771.30
R25/960/3	172.43	87.6	4997.29	**205.13**	204.68	204.96	0.19	38.25	**205.13**	204.33	204.82	0.37	2096.38
R25/240/4	126.02	19.7	2996.42	**130.04**	129.95	129.97	0.04	1.53	129.95	128.30	129.37	0.66	112.36
R25/480/4	175.47	69.4	4339.86	**197.36**	195.98	197.02	0.60	17.18	**197.36**	196.46	197.05	0.39	623.25
R25/720/4	201.27	60.7	5128.41	**222.08**	173.02	209.67	21.16	8.70	**222.08**	221.33	221.80	0.35	600.91
R25/960/4	193.57	67.1	935.92	**228.49**	228.04	228.38	0.19	28.22	**228.49**	227.71	228.11	0.34	745.46
R25/240/5	143.08	50.1	3918.87	**152.72**	152.36	152.54	0.18	2.37	**152.72**	152.36	152.58	0.18	106.43
R25/480/5	197.44	61.8	6487.38	**220.85**	220.61	220.79	0.10	6.89	220.45	219.56	220.14	0.31	519.73
R25/720/5	219.05	47.6	5706.35	**240.15**	240.03	240.06	0.05	13.86	240.10	239.89	240.01	0.08	611.18
R25/960/5	224.28	44.2	1834.97	**243.91**	243.83	243.88	0.03	26.37	**243.91**	243.15	243.59	0.25	446.86
Avg	*158.19*	*45.3*	*4107.03*	*173.03*	*169.31*	*171.98*	*1.61*	*15.41*	*172.99*	*172.20*	*172.71*	*0.32*	*516.41*

C.3.2 Detailed results of ILS-MCS on large instances (50, 75 and 100 Customers)

In Tables 12–14 we detail the results from the enhanced BBC and the ILS-MCS for instance with 50, 75 and 100 customers.

Table 12: Results of ILS-MCS on instances with 50 customers

Instance	Enhanced BBC			ILS-MCS					
	z_{lb}	z_{ub}	Gap (%)	Best Obj	Worst Obj	Avg. Obj	Std. Dev.	Avg. Time(s)	Gap (%)
C50/200/2	44.09	138.13	213.28	108.15	107.72	108.06	0.17	1.2	27.82
C50/400/2	73.85	237.28	221.32	201.62	195.56	200.05	2.26	6.7	18.61
C50/600/2	182.30	316.37	73.55	264.86	264.39	264.63	0.15	24.4	19.55
C50/800/2	29.01	380.45	1211.68	307.74	307.18	307.59	0.21	54.9	23.69
C50/200/3	26.88	187.12	596.24	150.04	149.68	149.94	0.14	2.4	24.80
C50/400/3	211.08	313.69	48.61	264.62	264.36	264.56	0.10	12.6	18.57
C50/600/3	259.97	406.89	56.52	337.96	337.44	337.77	0.18	42.3	20.46
C50/800/3	59.16	480.93	712.92	382.95	380.88	382.15	0.71	65.4	25.85
C50/200/4	-	228.75	-	188.00	187.83	187.97	0.07	4.0	21.70
C50/400/4	225.52	376.45	66.93	320.89	320.36	320.57	0.20	25.3	17.43
C50/600/4	-	515.52	-	397.70	397.32	397.46	0.13	62.5	29.70
C50/800/4	38.75	559.19	1343.03	441.89	441.50	441.65	0.14	93.3	26.61
C50/200/5	-	266.89	-	220.75	220.75	220.75	0.00	5.4	20.90
C50/400/5	-	430.46	-	366.39	365.66	366.02	0.24	22.5	17.61
C50/600/5	-	742.42	-	447.37	445.20	446.42	0.75	127.1	66.31
C50/800/5	-	843.70	-	490.29	489.57	489.84	0.29	223.0	72.24
Avg	*115.06*	*339.65*	454.41	*305.7*	*304.7*	*305.3*	*0.4*	*48.3*	28.24
R50/258/2	103.54	123.74	19.51	107.30	106.46	106.99	0.30	2.0	15.66
R50/516/2	109.97	226.13	105.64	189.38	189.38	189.38	0.00	7.1	19.40
R50/774/2	15.48	307.46	1886.84	239.13	238.13	238.58	0.38	39.9	28.87
R50/1032/2	30.26	371.31	1126.96	268.84	268.84	268.84	0.00	86.0	38.12
R50/258/3	114.01	178.21	56.31	150.20	149.93	150.13	0.10	4.9	18.70
R50/516/3	190.41	308.49	62.01	255.12	251.92	252.85	1.18	20.4	22.00
R50/774/3	48.61	400.70	724.31	312.30	310.65	311.71	0.59	88.2	28.55
R50/1032/3	58.79	467.92	695.90	340.75	339.51	340.01	0.44	129.4	37.62
R50/258/4	78.82	227.74	188.94	189.50	186.54	188.61	1.09	6.5	20.75
R50/516/4	82.50	374.80	354.28	312.05	311.76	311.99	0.12	28.8	20.13
R50/774/4	93.56	468.85	401.11	368.38	367.16	367.92	0.52	112.1	27.43
R50/1032/4	92.66	703.65	659.41	392.77	390.81	392.06	0.69	158.4	79.47
R50/258/5	-	266.05	-	224.14	220.05	223.12	1.55	9.6	19.24
R50/516/5	-	427.20	-	354.49	353.61	353.82	0.34	36.9	20.74
R50/774/5	-	703.56	-	410.01	408.70	409.23	0.47	82.7	71.92
R50/1032/5	80.74	703.65	771.50	431.01	429.76	430.51	0.47	104.3	63.44
Avg	*84.57*	*374.05*	542.52	*284.1*	*282.7*	*283.5*	*0.5*	*57.3*	33.25

- indicates the enhanced BBC did not find the lower bound within 2 hours.

Table 13: Results of ILS-MCS on instances with 75 customers

Instance	Enhanced BBC z_{lb}	z_{ub}	Gap (%)	ILS-MCS Best Obj	Worst Obj	Avg. Obj	Std. Dev.	Avg. Time(s)	Gap (%)
C75/350/2	-	271.70	-	219.83	217.77	219.01	1.01	3.8	24.06
C75/700/2	-	446.70	-	356.69	354.88	356.25	0.69	39.1	25.39
C75/1050/2	-	577.91	-	434.98	433.03	434.24	0.65	169.3	33.08
C75/1400/2	-	683.83	-	479.71	479.24	479.56	0.17	749.6	42.60
C75/350/3	-	359.38	-	302.49	299.20	301.55	1.23	9.4	19.18
C75/700/3	-	575.43	-	470.30	468.00	469.83	0.91	89.8	22.48
C75/1050/3	-	727.77	-	557.00	551.78	555.83	2.03	311.6	30.93
C75/1400/3	-	847.29	-	593.38	593.04	593.25	0.16	781.1	42.82
C75/350/4	-	437.10	-	368.72	366.57	368.07	0.79	14.4	18.75
C75/700/4	-	937.70	-	555.79	554.82	555.36	0.42	147.5	68.85
C75/1050/4	-	846.53	-	640.76	640.07	640.40	0.23	459.0	32.19
C75/1400/4	-	1294.79	-	668.10	665.28	666.79	1.08	1703.6	94.18
C75/350/5	-	504.96	-	427.50	426.99	427.19	0.19	30.0	18.21
C75/700/5	-	768.09	-	627.20	626.26	626.79	0.32	157.5	22.54
C75/1050/5	-	1283.00	-	705.73	704.03	704.87	0.54	623.6	82.02
C75/1400/5	-	1332.92	-	727.42	725.05	726.27	0.95	2528.5	83.53
Avg	-	*743.44*	-	*508.5*	*506.6*	*507.8*	*0.7*	*488.6*	41.30
R75/278/2	-	154.12	-	124.45	124.18	124.30	0.09	4.4	23.99
R75/556/2	-	270.45	-	223.43	222.43	222.91	0.35	24.9	21.32
R75/834/2	-	372.11	-	284.34	282.53	283.15	0.73	110.6	31.41
R75/1112/2	-	452.40	-	325.09	322.10	324.18	1.20	372.8	39.55
R75/278/3	-	210.80	-	177.83	175.87	177.01	0.65	8.7	19.09
R75/556/3	-	367.86	-	303.26	301.29	302.39	0.78	64.2	21.65
R75/834/3	-	489.32	-	381.66	379.13	380.34	0.87	288.2	28.65
R75/1112/3	-	586.47	-	424.51	423.49	424.13	0.46	392.5	38.28
R75/278/4	-	264.36	-	226.89	226.89	226.89	0.00	14.7	16.51
R75/556/4	-	449.14	-	377.69	377.19	377.54	0.19	134.6	18.97
R75/834/4	-	584.51	-	459.28	458.10	458.56	0.45	300.5	27.47
R75/1112/4	-	957.31	-	497.52	495.51	496.75	0.75	513.7	92.71
R75/278/5	-	316.03	-	266.78	264.08	265.82	0.96	27.8	18.89
R75/556/5	-	525.09	-	438.51	435.74	437.80	1.03	83.6	19.94
R75/834/5	-	664.38	-	517.31	516.69	516.95	0.21	269.4	28.52
R75/1112/5	-	1053.59	-	554.65	553.40	553.88	0.45	617.7	90.22
Avg	-	*482.37*	-	*349.0*	*347.4*	*348.3*	*0.6*	*201.8*	33.57

- indicates the enhanced BBC did not find the lower bound within 2 hours.

Table 14: Results of ILS-MCS on instances with 100 customers

Instance	Enhanced BBC			ILS-MCS					
	z_{lb}	z_{ub}	Gap (%)	Best Obj	Worst Obj	Avg. Obj	Std. Dev.	Avg. Time(s)	Gap (%)
C100/350/2	-	282.14	-	219.83	219.83	219.83	0.00	7.4	28.34
C100/700/2	-	469.54	-	371.88	362.47	369.52	3.56	96.5	27.07
C100/1050/2	-	623.79	-	466.37	462.30	465.33	1.53	387.6	34.05
C100/1400/2	-	750.86	-	531.65	523.44	526.61	3.05	1584.7	42.58
C100/350/3	-	377.11	-	304.83	304.56	304.65	0.10	15.8	23.78
C100/700/3	-	620.05	-	502.66	502.14	502.37	0.17	94.1	23.43
C100/1050/3	-	804.13	-	613.20	610.86	612.29	0.78	511.4	31.33
C100/1400/3	-	953.11	-	679.10	673.35	677.33	2.31	1548.9	40.72
C100/350/4	-	460.20	-	382.91	382.53	382.69	0.13	26.4	20.25
C100/700/4	-	744.68	-	607.71	605.11	606.57	0.88	279.9	22.77
C100/1050/4	-	950.79	-	729.87	728.25	728.98	0.57	736.4	30.43
C100/1400/4	-	1111.71	-	784.09	781.52	783.21	0.88	1238.9	41.94
C100/350/5	-	535.13	-	452.49	450.04	451.71	0.87	56.9	18.47
C100/700/5	-	852.83	-	702.86	702.37	702.53	0.18	290.4	21.39
C100/1050/5	-	1075.06	-	822.04	819.74	821.15	0.79	683.5	30.92
C100/1400/5	-	1707.74	-	866.58	863.01	864.97	1.16	2211.2	97.43
Avg	-	*769.93*	-	*564.9*	*562.0*	*563.7*	*1.1*	*610.6*	33.43
R100/278/2	-	180.14	-	144.69	144.16	144.49	0.18	20.4	24.68
R100/556/2	-	310.19	-	250.85	249.00	249.69	0.82	82.6	24.23
R100/834/2	-	420.90	-	326.99	325.77	326.38	0.55	270.7	28.96
R100/1112/2	-	520.01	-	379.14	374.82	378.08	1.64	627.7	37.54
R100/278/3	-	246.78	-	203.69	202.62	203.27	0.38	14.7	21.41
R100/556/3	-	418.53	-	348.02	345.95	347.01	0.78	96.4	20.61
R100/834/3	-	563.60	-	441.82	440.48	441.40	0.51	539.4	27.68
R100/1112/3	-	680.13	-	495.20	491.01	493.50	1.43	2033.0	37.82
R100/278/4	-	302.62	-	257.56	255.76	256.71	0.70	22.0	17.88
R100/556/4	-	515.62	-	431.93	430.76	431.38	0.44	245.3	19.53
R100/834/4	-	679.56	-	535.02	528.29	533.16	2.51	1267.7	27.46
R100/1112/4	-	1144.26	-	588.96	585.56	587.92	1.30	1804.5	94.63
R100/278/5	-	359.10	-	309.54	308.57	309.23	0.35	50.1	16.13
R100/556/5	-	602.95	-	504.46	501.65	503.13	1.22	247.9	19.84
R100/834/5	-	1118.34	-	613.41	610.93	612.08	0.92	581.0	82.71
R100/1112/5		1299.75		665.78	662.07	663.96	1.44	1086.0	95.76
Avg	-	*560.16*	-	*406.1*	*403.6*	*405.1*	*0.9*	*561.8*	37.30

- indicates the enhanced BBC did not find the lower bound within 2 hours.

References

Abramson M, Audet C, Couture G, Dennis, Jr J, Le Digabel S, Tribes C (2019) The NOMAD project. Software available at https://www.gerad.ca/nomad/.

Adulyasak Y, Cordeau JF, Jans R (2014) Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. INFORMS Journal on Computing 26(1):103–120.

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

Audet C, Hare W (2017) Derivative-free and blackbox optimization (Springer).

Camm JD, Magazine MJ, Kuppusamy S, Martin K (2017) The demand weighted vehicle routing problem. European Journal of Operational Research 262(1):151–162.

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

Chen HK, Hsueh CF, Chang MS (2009) Production scheduling and vehicle routing with time windows for perishable food products. Computers & Operations Research 36(7):2311–2319.

Chen L, Miller-Hooks E (2012) Optimal team deployment in urban search and rescue. Transportation Research Part B: Methodological 46(8):984–999.

Chitsaz M, Divsalar A, Vansteenwegen P (2016) A two-phase algorithm for the cyclic inventory routing problem. European Journal of Operational Research 254(2):410–426.

Codato G, Fischetti M (2006) Combinatorial Benders' Cuts for Mixed-Integer Linear Programming. Operations Research 54(4):756–766.

Coutinho WP, Fliege J, Battarra M (2018) Glider Routing and Trajectory Optimisation in disaster assessment. European Journal of Operational Research.

Edmonds J, Karp RM (1972) Theoretical improvements in algorithmic efficiency for network flow problems. Journal of the ACM (JACM) 19(2):248–264.

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

Erdoğan G, Laporte G (2013) The orienteering problem with variable profits. Networks 61(2):104–116.

Erkut E, Zhang J (1996) The maximum collection problem with time-dependent rewards. Naval Research Logistics 43(5):749–763.

Fermi E, Metropolis N (1952) Numerical solution of a minimum problem. Los Alamos Unclassified Report LA–1492, Los Alamos National Laboratory, Los Alamos, USA.

Fischetti M, González JJS, Toth P (1998) Solving the Orienteering Problem through Branch-and-Cut. INFORMS Journal on Computing 10(2):133–148.

Golden BL, Levy L, Vohra R (1987) The orienteering problem. Naval Research Logistics (NRL) 34(3):307–318.

Gunawan A, Lau HC, Vansteenwegen P (2016) Orienteering problem: A survey of recent variants, solution approaches and applications. European Journal of Operational Research 255(2):315–332.

Gunawan A, Ng KM, Kendall G, Lai J (2018) An iterated local search algorithm for the team orienteering problem with variable profits. Engineering Optimization 50(7):1148–1163.

Ibaraki T, Imahori S, Kubo M, Masuda T, Uno T, Yagiura M (2005) Effective local search algorithms for routing and scheduling problems with general time-window constraints. Transportation science 39(2):206–232.

Johnson SG (2019) The nlopt nonlinear-optimization package. http://github.com/stevengj/nlopt.

Kraft D (1994) Algorithm 733: Tomp–fortran modules for optimal control calculations. ACM Transactions on Mathematical Software (TOMS) 20(3):262–281.

Laporte G, Louveaux FV (1993) The integer L-shaped method for stochastic integer programs with complete recourse. Operations Research Letters 13(3):133–142.

Le Digabel S (2011) Algorithm 909: Nomad: Nonlinear optimization with the mads algorithm. ACM Transactions on Mathematical Software (TOMS) 37(4):44.

Lefever W, Aghezzaf EH, Hadj-Hamou K (2016) A convex optimization approach for solving the single-vehicle cyclic inventory routing problem. Computers & Operations Research 72:97–106.

Nayeri S, Asadi-Gangraj E, Emami S (2018) Metaheuristic algorithms to allocate and schedule of the rescue units in the natural disaster with fatigue effect. Neural Computing and Applications 1–21.

Pietz J, Royset JO (2013) Generalized orienteering problem with resource dependent rewards. Naval Research Logistics 60(4):294–312.

Powell MJD (1994) A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation, 51–67 (Springer Netherlands).

Rahmaniani R, Crainic TG, Gendreau M, Rei W (2017) The Benders decomposition algorithm: A literature review. European Journal of Operational Research 259(3):801–817.

Rauchecker G, Schryen G (2019) An exact branch-and-price algorithm for scheduling rescue units during disaster response. European Journal of Operational Research 272(1):352–363.

Runarsson TP, Yao X (2005) Search biases in constrained evolutionary optimization. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 35(2):233–243.

Schryen G, Rauchecker G, Comes T (2015) Resource Planning in Disaster Response. Business & Information Systems Engineering 57(4):243–259.

Svanberg K (2002) A class of globally convergent optimization methods based on conservative convex separable approximations. SIAM journal on optimization 12(2):555–573.

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

Thorsteinsson ES (2001) Branch-and-check: A hybrid framework integrating mixed integer programming and constraint logic programming. International Conference on Principles and Practice of Constraint Programming, 16–30.

Tsiligirides T (1984) Heuristic methods applied to orienteering. Journal of the Operational Research Society 35(9):797–809.

Vansteenwegen P, Gunawan A (2019) Orienteering Problems: models and algorithms for vehicle routing problems with profits. (Springer Nature).

Vansteenwegen P, Souffriau W, Oudheusden DV (2011) The orienteering problem: A survey. European Journal of Operational Research 209(1):1–10.

Vigerske S, Gleixner A (2018) SCIP: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. Optimization Methods and Software 33(3):563–593.

Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming 106(1):25–57.

Wex F, Schryen G, Feuerriegel S, Neumann D (2014) Emergency response in natural disaster management: Allocation and scheduling of rescue units. European Journal of Operational Research 235(3):697–708.

Yu J, Aslam J, Karaman S, Rus D (2015) Anytime planning of optimal schedules for a mobile sensing robot. International Conference on Intelligent Robots and Systems (IROS), 5279–5286.

Yu Q, Fang K, Zhu N, Ma S (2019) A matheuristic approach to the orienteering problem with service time dependent profits. European Journal of Operational Research 273(2):488–503.

Zhong Y, Aghezzaf EH (2011) Combining dc-programming and steepest-descent to solve the single-vehicle inventory routing problem. Computers & Industrial Engineering 61(2):313–321.

Audet C, Le Digabel S, Tribes C (2009) NOMAD user guide. Technical Report G–2009–37, Les cahiers du GERAD, https://www.gerad.ca/nomad/Downloads/user_guide.pdf.