The r -Interdiction Selective Multi-Depot Vehicle Routing Problem

Mir Ehsan Hesam Sadati^a, Deniz Aksen^{b,*}, Necati Aras^c

^a *Department of Industrial Engineering, İstanbul Kültür University, İstanbul, Turkey*

^b *College of Administrative Sciences and Economics, Koç University, İstanbul, Turkey*

^c *Department of Industrial Engineering, Boğaziçi University, İstanbul, Turkey*

E-mail: e.hesam136@gmail.com [Hesam Sadati]; daksen@ku.edu.tr [Aksen]; arasn@boun.edu.tr [Aras]

Revised and resubmitted on 03 February 2019

Abstract

The protection of critical facilities has been attracting increasing attention in the past two decades. Critical facilities involve physical assets such as bridges, railways, power plants, hospitals, and transportation hubs among others. In this study we introduce a bilevel optimization problem for the determination of the most critical depots in a vehicle routing context. The problem is modelled as an attacker-defender game (Stackelberg game) from the perspective of an adversary agent (the attacker) who aims to inflict maximum disruption on a routing network. We refer to this problem as the r -interdiction selective multi-depot vehicle routing problem (RI-SMDVRP). The attacker is the decision maker in the upper level problem (ULP) who chooses r depots to interdict with certainty. The defender is the decision maker in the lower level problem (LLP) who optimizes the vehicle routes in the wake of the attack. The defender has to satisfy all customer demand either using the remaining depots or through outsourcing to a third party logistics service provider. The ULP is solved through exhaustive enumeration, which is viable when the cardinality of interdictions does not exceed five among nine depots. For the LLP we implement a Tabu Search heuristic adapted to the selective multi-depot VRP. Our results are obtained on a set of RI-SMDVRP instances synthetically constructed from standard MDVRP test instances.

Keywords: Bilevel programming; interdiction; multi-depot vehicle routing problem; outsourcing; tabu search

1. Introduction and literature review

The protection of critical facilities has attracted much attention from the OR community in the past two decades. Critical facilities involve physical assets such as bridges, railways, terminals, power plants, hospitals, police stations, and transportation hubs among others (Church and Scaparra, 2007). Destruction or interdiction of these facilities causes disruption in the network and system operations. A man-made attack, i.e. intentional harm inflicted by an adversary agent is the main focus in interdiction models. Earliest network interdiction research can be attributed to Wollmer (1964 and 1970) who considered a capacitated network flow model with the interdiction of a fixed number of arcs in order to minimize the network flow capacity or to maximize the minimum cost flow. Israeli and Wood (2002) studied the problem of identifying the critical arcs in a shortest path network. Critical arcs are designated as those arcs which, in the case of (complete) interdiction, will cause the maximum increase in the shortest distance between two specific nodes. Arroyo and Galiana (2005) generalized the "terrorist threat problem" in a real-world electrical power grid system which was originally proposed by Salmeron et al.

* Corresponding author. Tel.: +90 (1) 338 1684, Fax.: +90 (1) 338 1653, Email: daksen@ku.edu.tr (D. Aksen)

(2004) to a bilevel programming problem. In Arroyo and Galiana's (2005) bilevel formulation, "the terrorist" interdicts power system components to cause a loss of load greater than or equal to a specified level, and aims to minimize the number of necessary interdictions to this end. "The defender" in turn tries to minimize the level of load not served by taking corrective actions. Interdicting facilities in supply chains can also be implemented by military planners. For example, McMasters and Mustin (1970) considered the problem of interdicting enemy's supply lines by a strike planner in order to reduce the maximum flow capacity from the source to the sink node in a network. The authors proposed an algorithm to obtain the best interdiction plan that would minimize the network flow capacity.

The first published paper involving node interdiction in supply chain networks is due to Church et al. (2004). The authors formulated two models: the r -interdiction median problem (RIM) and the r -interdiction covering problem (RIC). In median-type service networks, the objective is to minimize the total cost of satisfying customers' demand where the cost associated with a customer is defined as the demand-weighted distance between the customer location and the facility serving the customer. In covering-type networks, the objective is to minimize the total amount of customer demand that can be satisfied by a given number of opened facilities. Whether a customer is covered or not is based on the proximity of the customer location to opened facilities. If there exists at least one facility that is closer to the customer than a threshold distance, then the customer is said to be covered. The goal in the RIM and RIC models is to find among p existing facilities a subset of r facilities whose interdiction would lead to the maximum possible disruption in a median- and coverage-type service network, respectively. The seminal paper of Church et al. (2004) provided also an early survey on interdiction models.

In all these studies, the models are constructed from the attacker's viewpoint with the objective of maximizing disruption in networks. When the model is embedded into a bilevel structure, the attacker (the adversary agent) plays the role of decision maker in the upper level problem, and the defender becomes the decision maker in the lower level problem. This structure results in an attacker-defender game known as a Stackelberg game (Von Stackelberg, 1952). Recent interdiction models in the literature attempt to identify the best protection strategies for the defender against the attacker. In this case, the attacker and the defender simply switch roles within the bilevel structure. The protection decision of the defender moves to the upper level and the interdiction decision of the attacker is modelled in the lower level. For example, Church and Scaparra (2007) embed the protection decision into the RIM model which they proposed in their previous study (Church et al., 2004), and formulate the r -interdiction median problem with fortification (RIMF). RIMF involves a median-type network in which service delivery occurs at p facility sites. Out of p existing facilities, r will be lost to a man-made attack where $(q + r \leq p)$ holds true. The objective of this problem is to identify q facilities to be fortified in order to minimize the sum of demand-weighted post-attack shortest distances between surviving facilities and customers.

Aksen et al. (2010) modify the RIMF model by considering a budget constraint instead of a cardinality restriction on the protective resources. They also allow capacity expansion at surviving facilities so as to meet all customer demand in the wake of the attack. Scaparra and Church (2012) extend the RIMF model and consider the supply chain with capacitated facilities which can be subjected to attack. The authors formulate this problem as a tri-level optimization model. In the upper level, the defender (system planner) decides to protect some facilities in order to minimize the total cost in case of attack. In the middle level, the attacker interdicts r facilities to cause maximum disruption in the service delivery network. Finally in the lower level, the defender minimizes the total cost by assigning demands to the remaining facilities.

In this study, we introduce a bilevel optimization problem for the determination of the most critical depots in a vehicle routing problem. The problem is first modelled as an attacker-defender game from the perspective of an attacker who aims to inflict maximum disruption on a delivery or collection type routing network. We refer to this problem as the *r -Interdiction Selective Multi-Depot Vehicle Routing*

Problem (RI-SMDVRP). The attacker is the decision maker in the upper level problem (ULP). He chooses r out of m present depots to interdict. If targeted by the attacker, a depot is destroyed beyond repair with certainty and partial interdiction does not occur. The defender is the decision maker in the lower level problem (LLP) who optimizes the vehicle routes in the wake of the attack. The defender has to satisfy all customer demand either using the remaining depots or through outsourcing to a third party logistics service provider (3PL). In this regard, the defender selects which customers' demand to satisfy in-house using its own vehicle fleet and which ones' demand to have shipped by a 3PL from one of its unharmed depots.

MDVRP is an extension of the classical vehicle routing problem (VRP) where instead of a single depot there exist multiple depots and each customer can be visited by a vehicle dispatched from any of depots. MDVRP is known to be $\mathcal{NP}$ -hard (Lenstra and Rinnooy Kan, 1981). It can be represented as a graph theoretic problem. Let $G = (I, A)$ be a complete directed graph where I is the node set and A is the arc set. The node set I is divided into two subsets of n customer nodes and m depot nodes as $IC = \{1, \dots, n\}$ and $ID = \{n+1, \dots, n+m\}$, respectively. For each customer $i \in IC$ there exist a nonnegative demand quantity q_i and service time st_i . A nonnegative cost d_{ij} is associated with each arc $(i, j) \in A$ which implies the shortest path distance from node i to node j . We deal with problems where the distance matrix is symmetric, i.e. $d_{ij} = d_{ji}$ and satisfy the triangular inequality. The vehicle fleet in MDVRP consists of homogeneous vehicles of a fixed capacity Q . The objective is to design vehicle routes with the minimum total distance to be travelled by satisfying the following constraints: (i) Each customer is visited exactly once by a vehicle. (ii) Each vehicle starts and ends at the same depot. (iii) The load of a vehicle does not exceed its capacity. (iv) The total route duration (travel time plus service times) does not exceed the maximum tour duration if there exists such a temporal restriction.

MDVRP arises in many real-life case studies of logistics operations. Examples include, among others, recyclable waste collection systems studied in Lopes et al. (2015) and horizontal cooperation in urban distribution studied in Quintero-Araújo et al. (2016). Solution approaches for MDVRP can be divided into two categories: exact and approximate approaches. As mentioned before, being an $\mathcal{NP}$ -hard problem, it is difficult to find an optimal solution even for moderate size instances. Crevier et al. (2007) report that there exist few exact approaches for solving small size MDVRP instances. Laporte et al. (1984) had solved MDVRP to optimality for up to 50 customers by implementing a branch-and-bound (BB) method. Using exact approaches to obtain high quality solutions to large size instances requires relatively long computational times. Therefore, various approximate approaches including a tabu search algorithm (Cordeau and Maischberger, 2012), a local search method (Subramanian et al., 2013), and a genetic algorithm (Vidal et al., 2014) have been developed to find near-optimal solutions in reasonable times.

In RI-SMDVRP the defender has to satisfy the demand of each customer either by dispatching vehicles from surviving (non-interdicted) depots or by outsourcing it to a 3PL. The latter option may be preferred when the cost of visiting customers with vehicles is greater than the cost of outsourcing them. To capture that option in our model, we incorporate the outsourcing cost into the objective function of the defender in the LLP, and refer to this problem as the *Selective Multi-Depot Vehicle Routing Problem* (SMDVRP). The most relevant studies of this problem are due to Aras et al. (2011), Stenger et al. (2013) and Stenger et al. (2013). The first paper studies a profit-maximizing extension of the classical MDVRP with pricing where visiting a customer generates a net profit. The second paper considers an extension of the MDVRP where customers can be visited either by a vehicle of the private fleet based at owned depots or by vehicles of a subcontractor. The third paper addresses the single- and multi-depot versions of the prize-collecting vehicle routing problem with a non-linear cost which incorporates the option of outsourcing customers to subcontractors.

A potential application of the problem considered in our paper arises in the healthcare industry, in particular in the distribution of drugs in a metropolitan city. Every day drugs are transferred from drug depots to pharmacies to provide a quick and reliable service to clients. In fact, these depots act as a transshipment node between pharmacies and drug suppliers. The interdiction of one or several depots in a region would hinder the drug distribution process and put people in need of drugs in a difficult situation. In the event of interdiction, pharmacies previously visited by a vehicle the origin depot of which has been destroyed by the adversary agent should be served hereafter by a vehicle dispatched from an unharmed, thus still operational depot. Another potential application is in the military where ordnance has to be distributed from the armories to the training or combat zone. Clearly, the destruction of an armory by the enemy results in a degradation of the combat capability of the defender who owns and operates the armories.

The rest of the paper is organized as follows. Section 2 presents a polynomial-size mathematical programming formulation of RI-SMDVRP. Section 3 is dedicated to the detailed description of solution methods proposed for solving RI-SMDVRP. We basically solve the ULP through exhaustive enumeration, and develop a Tabu Search heuristic for the LLP. The results of our extensive computational study are reported in Section 4. Finally, conclusions and directions for future work are elaborated in Section 5.

2. Bilevel formulation of the RI-SMDVRP

In our problem RI-SMDVRP, the attacker and the defender seek to optimize the same objective function in opposite directions. The attacker decides which depots to interdict, whereas the defender tries to determine how to satisfy customers' demand as efficiently as possible in the post-attack stage. This problem fits into a bilevel programming framework where the attacker is the decision maker for the ULP and the defender is the decision maker for the LLP. In the sequel we propose a polynomial-size mixed-integer linear programming (MILP) formulation of the RI-SMDVRP as follows.

Index sets:

- $IC = 1, \dots, n$: The set of n customer nodes.
- $ID = n + 1, \dots, n + m$: The set of m depot nodes.
- $I = IC \cup ID$: The set of all nodes.

Parameters:

- d_{ij} : Distance (cost of traveling) from node $i \in I$ to node $j \in I$.
- c^v : Operating cost per vehicle.
- c^o : Outsourcing cost per unit demand.
- Q : Vehicle capacity.
- q_i : Demand or load to be delivered or to be collected from customer node $i \in IC$.
- r : Number of depots that can be interdicted by the attacker
- t_{ij} : Travel time from node $i \in I$ to node $j \in I$.
- st_i : Service time at node $i \in IC$.
- D_{max} : Maximum tour duration.
- M : Big- M value. It can be set to $D_{max} + \max_{(i,j) \in IC} \{st_j + t_{ij}\}$.

Decision variables:

X_{ij} : Binary variable indicating if arc $(i, j) \in A$ is traversed by a vehicle.

$H(W)$: Total cost of satisfying all customers' demand (i.e. the objective value).

W_j : Binary variable indicating that if depot $j \in ID$ is interdicted.

F_{ij} : Amount of goods flowing on arc $(i, j) \in A$ after collecting the “demand” at i .

Y_{ij} : Binary variable indicating the assignment of customer node $i \in IC$ to depot $j \in ID$.

A_i : Arrival time at node $i \in I$ where $A_j = 0$ for $j \in ID$ by convention.

RI-SMDVRP :

$$\text{Maximize } H(W) \quad (1)$$

$$\text{subject to: } \sum_{j \in ID} W_j = r \quad (2)$$

$$W_j \in \{0,1\} \quad j \in ID \quad (3)$$

$$\text{where } H(W) = \text{Minimize } \sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} d_{ij} X_{ij} + c^v \sum_{j \in ID} \sum_{i \in IC} X_{ji} + c^o \sum_{i \in IC} q_i (1 - \sum_{j \in ID} Y_{ij}) \quad (4)$$

$$\text{subject to: } \sum_{\substack{j \in I \\ i \neq j}} X_{ji} \leq 1 \quad i \in IC \quad (5)$$

$$\sum_{\substack{j \in I \\ i \neq j}} X_{ij} \leq 1 \quad i \in IC \quad (6)$$

$$\sum_{i \in IC} q_i Y_{ij} \leq Q \sum_{i \in IC} X_{ji} \quad j \in ID \quad (7)$$

$$\sum_{i \in ID} \sum_{j \in ID} X_{ij} = 0 \quad (8)$$

$$\sum_{j \in ID} Y_{ij} \leq 1 \quad i \in IC \quad (9)$$

$$X_{ij} \leq Y_{ij} \quad i \in IC, j \in ID \quad (10)$$

$$X_{ji} \leq Y_{ij} \quad i \in IC, j \in ID \quad (11)$$

$$X_{ik} + Y_{ij} - Y_{kj} \leq 1 \quad i \in IC, k \in IC, i \neq k, j \in ID \quad (12)$$

$$X_{ik} + Y_{kj} - Y_{ij} \leq 1 \quad i \in IC, k \in IC, i \neq k, j \in ID \quad (13)$$

$$\sum_{\substack{j \in I \\ j \neq i}} F_{ij} - \sum_{\substack{j \in I \\ j \neq i}} F_{ji} = q_i \sum_{k \in ID} Y_{ik} \quad i \in IC \quad (14)$$

$$F_{ij} \geq q_i X_{ij} \quad i \in I, j \in I, i \neq j \quad (15)$$

$$F_{ij} \leq (Q - q_j) X_{ij} \quad i \in I, j \in I, i \neq j \quad (16)$$

$$\sum_{\substack{i \in I \\ i \neq j}} X_{ij} - \sum_{\substack{i \in I \\ i \neq j}} X_{ji} = 0 \quad j \in I \quad (17)$$

$$X_{ij} + X_{ji} \leq 2 - \left\lceil \frac{(q_i + q_j)}{Q} \right\rceil \quad i \in IC, j \in IC, i < j \quad (18)$$

$$\sum_{\substack{j \in I \\ i \neq j}} X_{ij} + \sum_{\substack{j \in I \\ i \neq j}} X_{ji} = 2 \sum_{k \in ID} Y_{ik} \quad i \in IC \quad (19)$$

$$Y_{ij} \leq 1 - W_j \quad i \in IC, j \in ID \quad (20)$$

$$X_{ij} \in \{0,1\} \quad i \in I, j \in I \quad (21)$$

$$Y_{ij} \in \{0,1\} \quad i \in IC, j \in ID \quad (22)$$

$$F_{ij} \geq 0 \quad i \in I, j \in I \quad (23)$$

In the above formulation, Eqs. (1)–(3) represent the ULP and Eqs. (4)–(23) correspond to the LLP. Expression (1) shows the attacker's objective function. The first component in this objective is the total distance traveled by vehicles, the second component is the total vehicle operating cost, and finally the third component represents the outsourcing cost incurred due to unsatisfied demand. Constraint (2) shows that only r depots can be interdicted by the attacker. The binary constraints on the attacker's decision variables are in (3).

We refer to defender's problem LLP as the SMDVRP the objective function of which in (4) is the same as in (1). Incoming and outgoing degree constraints (5) and (6) ensure that a customer can be visited at most once. Constraints (7) indicate the minimum number of vehicles required. Constraint (8) makes sure that there is no vehicle traveling between two depots. Constraints (9) satisfy the requirement that a customer can be assigned to at most one depot. Constraints (10) and (11) indicate logical constraints stating that no vehicle can go from customer i to depot j or vice versa unless customer i is assigned to depot j . Constraints (12) and (13) together enforce that if two customers i and k are visited consecutively by a vehicle, then they must be assigned both to the same depot j . Constraints (14)–(16) indicate Gavish-Graves single commodity flow constraints which serve as subtour elimination constraints. The cardinality of these constraints is polynomial in n and m (see Gavish and Graves, 1978). Constraint (14) indicates the flow conservation at each customer node while constraints (15) and (16) impose tight bounds on the flow variables F_{ij} . Eqs. (15) indicate that goods flowing between nodes i and j should be greater than or equal to the demand of node i if a vehicle travels from i to j . Eqs. (16) ensure that if a vehicle travels from location i to location j (depot or customer node), then the amount of goods flowing from node i to node j after collecting the demand at node i cannot be greater than $Q - q_j$. Constraints (17) ensure that each customer as well as depot node has the same number of outgoing and incoming arcs.

Constraints (18) correspond to valid inequalities eliminating subtours of size two. They are derived from the exponential number of subtour elimination constraints proposed for the traveling salesman problem (TSP) in Dantzig et al. (1954). Their extended generalized version is provided in Aksen et al. (2018) as $\sum_{i \in S} \sum_{\substack{j \in S \\ j \neq i}} X_{ij} \leq |S| - L(S) \quad S \subseteq IC; S \neq \emptyset; |S| \geq 2$. In these constraints $L(S)$ can be written as

$$L(S) = \left\lceil \frac{q(S)}{Q} \right\rceil \quad \text{where } q(S) \text{ is equal to the total demand in subset } S. \text{ Now if our subset } S = \{i, j\}$$

contains two nodes i and j , i.e. if $|S| = 2$, then we can write the original constraints as in (18). They prevent the formation of subtours of size two. We remark that these valid inequalities are redundant in the presence of (14)–(16); however, they tighten up the SMDVRP formulation. Their substantial benefit

in the single commodity flow formulation of the capacitated vehicle routing problem (CVRP) has been empirically demonstrated in Aksen et al. (2018).

Constraints (19) are another set of valid equalities stating that a customer node when assigned to a depot must be visited by some vehicle.

Constraints (20) link the LLP with the ULP in that no customers can be assigned to an interdicted depot. They are written deliberately in a disaggregated fashion for each customer i and depot j combination. Finally, constraints (21) and (22) force the defender's routing and assignment variables to be binary while constraints (23) are nonnegativity constraints on the flow variables.

Most standard MDVRP test instances involve a maximum tour duration constraint (MTD). It implies that the total travel time of each vehicle tour is restricted by a maximum tour duration D_{max} . At each customer node $i \in IC$ there is a customer dependent service time st_i . The travel time t_{ij} between each pair of nodes i and j is also known a priori. In some test instances service times are ignored ($st_i = 0$), in which case MTD becomes the maximum tour length constraint. To handle this constraint in the MDVRP (or SMDVRP) model, we introduce nonnegative continuous arrival time variables A_i and equations (24)–(27) as follows:

$$A_j \geq A_i + st_i + t_{ij} - M(1 - X_{ij}) \quad i \in I, j \in IC, i \neq j \quad (24)$$

$$A_j \leq A_i + st_i + t_{ij} + M(1 - X_{ij}) \quad i \in I, j \in IC, i \neq j \quad (25)$$

$$A_i + st_i + \sum_{j \in ID} t_{ij} X_{ij} \leq D_{max} \quad i \in IC \quad (26)$$

$$A_i \geq 0 \quad i \in IC \quad (27)$$

Note that depot arrival times are all fixed to zero, i.e. $A_j = 0 \quad j \in ID$. Constraints (24) and (25) together state that the arrival time A_j at node j is equal to $A_i + t_{ij} + st_i$ if a vehicle travels from node i to node j . Eqs. (26) enforce MTD for each node $i \in IC$. They could be written also in a nonconsolidated manner as shown in (28). Finally, constraints (27) stand for the nonnegativity of arrival time variables A_i .

$$A_i + st_i + t_{ij} X_{ij} \leq D_{max} \quad i \in IC, j \in ID \quad (28)$$

3. Three solution methods for RI-SMDVRP

Moore and Bard (1990) proved that mixed-integer bilevel programming problems are $\mathcal{NP}$ -hard. In regard to the $\mathcal{NP}$ -hardness of RI-SMDVRP, we first need to show that a generic instance of the TSP with Profits (TSPP) is polynomial-time reducible to RI-SMDVRP, i.e. $TSPP \leq_p RI-SMDVRP$ (Kleinberg and Tardos, 2006). For a given TSPP instance, we can duplicate its depot and superpose the new depot at the same location with the original one. We can set the interdiction cardinality r to 1. Furthermore, D_{max} can be set to infinity, and Q can be set to the number of customers n . We can assume a unit demand for each customer $i \in IC$. By doing so, we obtain an RI-SMDVRP instance with $m = 2$ depots where an arbitrary one is to be interdicted. This concludes that $TSPP \leq_p RI-SMDVRP$, i.e. TSPP is a special case of RI-SMDVRP. Clearly, solving this special case problem is as hard as solving the generic TSPP. Since the class of TSPP is already proven to be $\mathcal{NP}$ -hard (Feillet et al., 2005), we deduce that RI-SMDVRP is also an $\mathcal{NP}$ -hard problem.

Due to the $\mathcal{NP}$ -hardness of RI-SMDVRP, heuristic techniques could prove a viable approach for its solution. Note that the ULP in RI-SMDVRP consists of only one set of decision variables which keep track of depots to be interdicted. We solve ULP through exhaustive enumeration (EE) which is viable when in the real application the cardinality of interdictions does not exceed five among nine depots. EE explores $\binom{m}{r}$ combinations of interdicting r depots among m depots and in each combination (i.e. in each interdiction pattern), the attacker strikes r depots, and then the defender optimizes the vehicle routes. For the optimization of the LLP (i.e. the corresponding SMDVRP) we employ a tabu search heuristic (TSH).

We propose three methods for solving RI-SMDVRP where in the second and third method we improve the first method's solution quality by using CPLEX. In METHOD 1, for each interdiction pattern we solve the related LLP with TSH and find the best interdiction pattern, i.e. the one which yields the highest LLP objective value. In METHOD 2, we first obtain the best interdiction pattern of METHOD 1, and then solve the ensuing LLP one more time with CPLEX using the TSH solution of METHOD 1 as the initial feasible solution. In METHOD 3, we compare the objective value provided by METHOD 2 against that of the second best interdiction pattern—the one for which TSH finds the second highest LLP objective value. If it is greater, we stop. The solutions of METHOD 3 and METHOD 2 are considered to be equivalent since it becomes obvious that none of the remaining interdiction patterns would be able to produce a higher, thus better LLP objective for the attacker. Otherwise, for each of the remaining interdiction patterns whose TSH objective value is greater than the solution of METHOD 2, we solve the ensuing LLP one more time with CPLEX in the expectation of finding a better interdiction pattern.

METHOD 1 is the fastest method with the lowest accuracy. METHOD 2 is the medium fast method with high accuracy. Finally, METHOD 3 is potentially the slowest method with the highest accuracy. The detailed description of the three methods is given below.

METHOD 1

- 1: **Loop Over** all interdiction patterns $IP(p)$ produced in the **EE** algorithm for the solution of the ULP
- 2: For each interdiction pattern p , solve the ensuing LLP using the **TSH**;
- 3: Record the best interdiction pattern p^{best} that yields the highest LLP objective value*;
- 4: **End Loop**
- 5: Report p^{best} and the best SMDVRP solution associated with it;**

* The highest LLP objective value implies the best objective value for the attacker.

** The best SMDVRP solution implies the least-cost routing and outsourcing plan for the defender.

METHOD 2

- 1: **Loop Over** all interdiction patterns $IP(p)$ produced in the **EE** algorithm for the solution of the ULP
- 2: For each interdiction pattern p , solve the ensuing LLP using the **TSH**;
- 3: Record the best interdiction pattern p^{best} and the associated routing plan ro^{best} that yields the highest LLP objective value;
- 4: **End Loop**
- 5: For the interdiction pattern p^{best} , solve the ensuing LLP one more time with **CPLEX*** using ro^{best} as the initial feasible solution of the MILP problem, and obtain ro_Cplex **;
- 6: Report p^{best} along with ro_Cplex ;

* Subject to a time limit of three hours.

** From the defender's cost minimization perspective, ro_Cplex will be at least as good a routing plan as ro^{best} .

METHOD 3

- 1: **Loop Over** all interdiction patterns $IP(p)$ produced in the **EE** algorithm for the solution of the ULP
- 2: For each interdiction pattern p , solve the ensuing LLP using the **TSH** and record its objective value in an array of objective values denoted by $Z_TSH[]$ along with the index of the associated pattern p ;
- 3: **End Loop**
- 4: Sort the array $Z_TSH[]$ in descending (nonincreasing) order*;
- 5: For the interdiction pattern stored in $Z_TSH[1]$, solve the ensuing LLP one more time with **Cplex** using the routing plan ro stored in $Z_TSH[1]$ as the initial feasible solution of the MILP problem, and obtain $ro_Cplex[1]$ with the objective value stored in $Z_Cplex[1]$;
- 6: Set $Z^{best} = Z_Cplex[1]$, $p^{best} = p$ implied in $Z_Cplex[1]$, and $ro^{best} = ro$ implied in $Z_Cplex[1]$;
- 7: **If** $Z_Cplex[1] \geq Z_TSH[2]$
- 8: **STOP!** *// It is now obvious that none of the remaining interdiction patterns [2] through $\binom{m}{r}$ can produce a higher LLP objective for the attacker.*
- 9: **Else**
- 10: **Loop Over** all interdiction patterns p for which $Z^{best} < Z_TSH[p]$
- 11: Solve the ensuing LLP one more time with **Cplex** using ro implied in $Z_TSH[p]$ as the initial feasible solution of the MILP problem, and obtain $ro_Cplex[p]$ with the objective value $Z_Cplex[p]$;
- 12: **If** $Z_Cplex[p] > Z^{best}$
- 13: Update $Z^{best} = Z_Cplex[p]$, $p^{best} = p$ implied in $Z_Cplex[p]$, and $ro^{best} = ro$ implied in $Z_Cplex[p]$;
- 14: **End If**
- 15: **End Loop**
- 16: **End If**
- 17: Report p^{best} along with ro^{best} and associated LLP objective value Z^{best} ;

* $Z_TSH[1]$ contains the costliest interdiction pattern p^{best} , hence the best one for the attacker, i.e. the one with the highest LLP objective. On the other hand, $Z_TSH[\binom{m}{r}]$ contains the least costly interdiction pattern with the least degree of disruption on the MDVRP network.

In all three methods, we solve for each interdiction pattern p in Step 2 the ensuing LLP (the SMDVRP) using TSH. The description of TSH is provided in the next subsection.

3.1 Tabu Search heuristic (TSH)

Tabu Search (TS), a trajectory based heuristic search method, was originally proposed by Glover (1986). The fundamental principle of TS is to trace local search whenever it coincides with a local optimum by permitting non-improving moves. Tabu list is used to prevent cycling over a sequence of solutions (iterations). This process is done by forbidding the process to go back to previously visited solutions. It requires a long term memory and excessive recording. Instead, a short term memory can be used by declaring some move elements as tabu. The tabu elements of a move cannot be revisited, reversed or reused for the next λ iterations. This duration of tabu status is referred to as the tabu tenure.

Our TSH algorithm is accompanied by strategic oscillation (allowing capacity- or time-infeasible solutions). Some of the explored neighborhood structures can be infeasible with respect to vehicle capacity or maximum tour duration constraints giving rise to capacity- or time-infeasible solutions. We accept infeasible solutions after penalizing their objective values by an amount proportional to the

magnitude of infeasibility. We also implement other common features in our TSH such as intensification and diversification. Intensification consists of spending more time around the best known solutions through greedy local search, while diversification ensures that the search process will not be restricted to a limited portion of the solution space. The components of our TSH algorithm are in the following subsections.

3.1.1 Initial solution generation method

For the generation of an initial solution we employ the classical Clarke and Wright savings heuristic (CWH) which was originally proposed for the VRP in (Clarke and Wright, 1964). We adapted it to the SMDVRP by complementing with the so-called 1-node and 2-node iterative marginal cost analysis (iMCA). An earlier marginal profit-driven version of iMCA was proposed in (Aksen and Aras, 2006) for solving VRPs with customer selection and profit maximization.

The resultant method is performed in two phases. In the first phase we solve the given SMDVRP as a standard MDVRP by using CWH followed by a rich neighborhood search procedure. At the beginning of the second phase, we calculate the marginal cost of each customer according to the set of routes constructed in the first phase. We then apply 1-Node and 2-Node iMCA where we decide to keep or to remove individual customer nodes or chains of two customer nodes based on their marginal costs. Removal of customer nodes necessitates that their demands be outsourced. It also requires that the marginal costs of the predecessor and successor nodes be recalculated. This inspection is repeated until all customer nodes remaining in the CWH solution have a strictly negative marginal cost. The pseudo code of the 1-Node iMCA is presented in Table A1 in Appendix A. We refer to this greedy heuristic method as CWH+iMCA. Its pseudo code is presented in Table 1. Note that Step 5: in the pseudo code is necessary because even though the marginal cost of every customer on a given route is negative, it is possible that its total cost is greater than its outsourcing cost (i.e. unit outsourcing cost multiplied by the total demand served on the route). Once such a route is detected, we have to outsource all customers on that route and discard it from the solution. The remaining steps of the algorithm are described below.

Table 1
The generation of an initial solution for TSH using CWH+iMCA

1:	Solve the MDVRP with the current set of customers and operational depots using classical Clarke and Wright savings heuristic (CWH);
2:	Implement Neighborhood Search Procedure (NSP) on the set of routes obtained in Step 1;
3:	Apply first 1-Node, then 2-Node iMCA until the marginal cost of each remaining customer is negative;
4:	If neither 1-Node nor 2-Node iMCA discards customers from the current routes, then go to Step 5. Otherwise (i.e., some customers are discarded), go to Step 1;
5:	For each route still present in the solution: If $\text{Cost of Route} \leq (\text{Outsourcing Cost} \times \text{Total demand of Route})$ STOP! Else remove the route and discard all customers on it from the solution;

CWH is one of the most widely-used heuristics for generating an initial solution of a routing problem. It is based on the *savings* notion and works well for both directed and undirected problems. First, each customer is placed exclusively on a dedicated route to begin with a feasible solution comprised of as many routes as the number of customers. Then, for all pairs of customers a savings value is calculated. Any two routes are merged to a new single route on the basis of the corresponding savings provided that the new route is feasible with respect to capacity and time constraints. There are two versions of CWH:

Parallel version and sequential version. Laporte et al. (2000) demonstrated that the parallel version produces better solutions than the sequential one on CVRP-CMT benchmark instances. In the parallel version, savings values are first sorted from highest to lowest. Routes are then selected for merger starting with that pair of routes whose exterior nodes promise the highest savings.

CWH can be extended to solve a given MDVRP as follows: Each customer is initially assigned to the nearest depot. For any depot and its assigned customers, we implement the classical CWH. Thereafter, to improve the solution quality of CWH we implement a rich neighborhood search procedure (NSP) that entails the following neighborhood structures.

3-Opt and Or-Opt:

Due to their complexity these structures are explored within the same route. Only intraroute moves are considered.

1-0 Move, 2-0 Move, 1-1 Exchange, 2-2 Exchange, and 2-Opt:

These structures are explored both within the same route and on two different routes.

When a given neighborhood structure is explored within the same route, we resort to the best improvement (steepest descent) strategy for the selection of the next solution. In this strategy the current neighborhood is explored exhaustively, and all possible moves are tried for a solution to select (Talbi, 2009). If interroute moves are also allowed in a neighborhood structure, we resort to the first improvement strategy which consists in choosing the first improving neighbor that is better than the current solution. That improving neighbor is immediately selected to replace the current solution. As pointed out in Talbi (2009), the first improvement strategy performs a partial evaluation of the neighborhood, thus is more time-efficient in large neighborhoods entailing two different routes.

Note that the set of routes obtained with CWH and NSP cover all the customers. Upon these routes we apply 1-Node and 2-Node iMCA to improve the solution quality further, i.e. to decrease the total cost objective. The iMCA procedure finds customers that should be outsourced, hence dismissed from the routing plan. It calculates for each customer its marginal cost. A customer is either dropped from the solution for good or visited by a vehicle based on its marginal cost. A discussion of the 1-Node and 2-Node iMCA procedures can be found in Appendix A.

3.1.2 Neighborhood structures

The neighborhood structures of TSH in our implementation are 1-0 Move, 1-1 Exchange, 2-Opt, 1-Add, 2-Add, 1-Drop and 2-Drop. The first three neighborhoods are called routing moves and the last four neighborhoods are called customer selection moves. Customer selection moves are explained below.

1-Add: Given a customer node not included in the routes yet, all possible insertion positions are checked, and the node is inserted in the best possible position. This position can be either the one causing the highest decrease in the current solution's objective value (routing and outsourcing cost), or if there exists no such position, it is the one which increases the objective value the least (cheapest insertion rule).

2-Add: Given a chain of two customer nodes not included in the routes, all possible insertion positions (same as in 1-Add) are checked, and the nodes are inserted in the best possible position consecutively.

1-Drop: Given a customer included in a route, it is removed from its current position. Subsequently, the predecessor and successor of this customer are connected. The rule of removing the customer is similar to the iMCA concept, i.e. customers with positive marginal cost are primary candidates for removal.

2-Drop: Given a customer and its successor visited in a tour, they are removed from their route. After removing these two consecutive nodes, their predecessor and successor are connected. The removing rule is defined as in the explanation of the 1-Drop move.

Note that at each iteration of TSH the execution of 1-Drop (2-Drop) move may eliminate only one customer (two consecutive customers) among all customers visited in the current routes. Likewise, 1-Add (2-Add) move inserts at most one (two) customer nodes at each iteration of TSH.

3.1.3 Granular neighborhoods

TSH algorithm needs to explore all mentioned neighborhood structures at each iteration in order to obtain a high quality solution. This exploration inevitably consumes long computing times. One of the most successful approaches developed to decrease the computing time in tabu search for routing problems is due to Toth and Vigo (2003). They proposed granular neighborhoods which speeds up TSH significantly. The basic idea is to explore only a subset of promising moves eliminating a great number of unpromising moves. In our implementation we capitalize on the same idea with some modifications. We accept an arc to be considered in a move (to be one of the admissible arcs in that move) if its cost (distance) is less than a granularity threshold value defined as $\varphi = \frac{Z}{(n+v)}$ where Z is the objective value

of the CWH+iMCA solution, n stands for the number of customers, and v stands for the number of routes in the solution. The admissible arcs are rebuilt from scratch every $3n$ iterations. In this case, the granularity threshold will be $\varphi' = \frac{Z'}{(n'+v')}$ where Z' is the objective value obtained by TSH after $3n$

iterations, n' and v' stand for the number of customers and routes in the solution after $3n$ iterations, respectively.

3.1.4 Tabu conditions

TSH algorithm employs tabu conditions to prevent cycling and to enforce searching various regions of the solution space. To this end, tabu conditions are created for each neighborhood structure after updating the current solution. The tabu condition is revoked at the end of tabu duration λ . In our implementation tabu duration also known as the tabu tenure is set to $\lambda = \lceil 7.5 \log_{10} n \rceil$ (Cordeau et al., 1997). The tabu conditions of the neighborhood structures are as follows:

- 1-0 Move:* A customer whose position is changed by *1-0 Move* cannot be dislocated by the same move.
- 1-1 Exchange:* Customers whose positions are swapped by *1-1 Exchange* cannot be re-swapped by the same move.
- 2-Opt:* Customers who were moved by *2-Opt* cannot enter another *2-Opt* move.
- 1-Add:* The added customer cannot be dropped from the routing plan.
- 2-Add:* The added chain of two customers cannot be dropped from the routing plan.
- 1-Drop:* The dropped customer cannot be added to the routing plan.
- 2-Drop:* The dropped chain of two customers cannot be added to the routing plan.

3.1.5 Aspiration criterion

Our TSH algorithm has one aspiration criterion which consists of revoking tabu conditions for a move if it results in a better (smaller) objective value than the current incumbent. In this case we override the tabu condition and perform the move.

3.1.6 Penalty value for strategic oscillation (Allowing infeasible solutions)

In TSH algorithm, some solutions generated during the exploration of the neighborhoods may turn out infeasible with respect to vehicle capacity and maximum tour duration constraints. In such cases TSH computes a new objective value that includes the original objective value as well as penalties associated with the infeasibilities. For a given solution S the new objective function with penalties is expressed as:

$$Z'(S) = Z(S) + \alpha Q(S) + \beta D(S) \quad \text{where} \quad Z(S) = \sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} d_{ij} X_{ij} + c^v \sum_{j \in ID} \sum_{i \in IC} X_{ji} + c^o \sum_{i \in IC} q_i (1 - \sum_{j \in ID} Y_{ij}),$$

$$Q(S) = \sum_{j \in ID} \left[\sum_{i \in IC} q_i Y_{ij} - Q \sum_{i \in IC} X_{ji} \right]^+ \quad \text{and} \quad D(S) = \sum_{i \in IC} \left[A_i + st_i + \sum_{j \in ID} t_{ij} X_{ij} - D_{max} \right]^+. \quad (29)$$

$Q(S)$ and $D(S)$ represent the violation in capacity and time constraints, receptively. α and β are positive parameters, and $[\cdot]^+ = \max\{0, \cdot\}$. If a given solution S is feasible, then $Z'(S) = Z(S)$. Adjusting the values of the penalty parameters is important because choosing too high penalty parameter values prevents the algorithm from visiting infeasible solutions, whereas too low penalty parameter values will fall short of detecting feasible solutions on the TSH trajectory. In this regard, the best option is to go with an adaptive update mechanism in which penalty parameters are to be updated after each iteration in order to find feasible and infeasible solutions as much equally often as possible. In our implementation, if the current solution S_{cur} after each iteration is capacity- or time-infeasible, we set $\alpha = \alpha(1 + \delta)$ ($\beta = \beta(1 + \delta)$); if it is capacity or time-feasible, we set $\alpha = \alpha / (1 + \delta)$ ($\beta = \beta / (1 + \delta)$). A reasonable value for δ is 0.5 as suggested in Cordeau et al. (1997). The initial values of α and β are both set to 1 in the first iteration.

3.1.7 Intensification and diversification

The goal of intensification is to investigate more deeply those regions of the search space in which a new incumbent solution is found. This is so because they could have the potential of hosting even better incumbents (Gendreau, 2013). In our algorithm, we apply intensification whenever a new incumbent is obtained. Once a new incumbent S^* is obtained, a series of neighborhood (local) search procedures, i.e. NSPs are applied to it in the hope of improving its objective value further. After applying intensification, TSH algorithm sometimes finds a local optimal solution that is far from the global optimal. This is due to the fact that TSH may spend more time in a restricted region and thus fail to investigate the most fertile regions of the search space. So diversification helps to diminish this issue by forcing the search into previously unexplored regions (Gendreau, 2013). In our tabu search implementation we use the same diversification technique that was proposed by Gendreau et al. (2008) to escape from local optimal solutions.

Let S_{cur} be the current solution and S_{new} the new solution obtained by one of the neighborhood structures $N(S)$. If $Z'(S_{new}) \geq Z'(S_{cur})$, then the penalty term $\eta = \sqrt{nv} \left(\frac{\rho_{i_{N(S)}}}{t} \right) (Z'(S_{new}) - Z'(S_{cur}))$ is

added to $Z'(S_{new})$. In this penalty term formula, the number of customers visited and the number of routes in S_{cur} are denoted by n and v , respectively. $\rho_{i_{N(S)}}$ indicates the number of times customer i is moved by the neighborhood structure $N(S)$, and t stands for the total number of iterations performed so far.

3.1.8 Termination criteria

We terminate the algorithm as soon as one of the following termination criteria is met.

- A fixed number of iterations are completed (we set it to 15,000),
- A certain number of non-improving iterations are observed.

The second condition depends on the topology of the problem instance. For a small size instance we can increase the number of iterations with non-improving solutions. This is due the fact that implementing rich neighborhood structures does not require high computational effort in small size problems, but in large problems spending more time on non-improving solutions is not rational. For MDVRP and SMDVRP we set this number as specified in Table 2. Consequently, the steps of our TSH algorithm for solving the LLP (SMDVRP) can be listed as in Table B1 in Appendix B. The details and explanatory illustrations of the routing neighborhood structures implemented in the TSH are presented in Appendix C which can be found in the electronic supplemental companion of the paper.

Table 2
Termination criteria of the TSH algorithm for MDVRP and SMDVRP

# of customers n	Maximum # of non-improving iterations		# of customers n	SMDVRP
	MDVRP			
$n \leq 50$	2000		$n \leq 50$	1500
$50 < n < 100$	300		$50 < n \leq 100$	300
$100 \leq n < 200$	200		$100 < n$	100
$200 \leq n$	100		–	–

The reduction concept presented in METHOD 3 (i.e. reducing the number of the CPLEX calls in lines 7 through 16) can be adapted to METHOD 1 as well in order to make it even faster without sacrificing its solution quality. To this end, we substitute the TSH with CWH+iMCA and CPLEX with TSH in METHOD 3. We refer to this updated method as METHOD 1'. In the next section we report the computational results obtained by this new reduced version of METHOD 1. We remark that METHOD 1' is expected to make fewer TSH calls compared to the original METHOD 1 outlined above. This is so because METHOD 1' may not check all possible interdiction patterns produced by EE for the solution of the ULP. This way, METHOD 1' has the potential to tackle problems with a larger number of depots than METHOD 1 within the same CPU time limit.

4. Computational results

4.1. Performance of TSH on MDVRP and SMDVRP

We performed our experiments on a Dell Precision T7810 model PC equipped with Intel Xeon® E5-2690 v4 2.60 GHz processor and 32 GB RAM running 64-bit Windows 7 Professional. Our algorithms were coded in C# of Microsoft® Visual Studio 2017. We employed IBM CPLEX version 12.7.1 as the commercial MILP solver called from inside C#.

In order to see the performance and accuracy of TSH on MDVRP instances before we applied it to RI-SMDVRP, we solved 33 standard MDVRP test instances which are downloadable from the portal VRP Web (<http://neo.lcc.uma.es/vrp>). We compared the TSH solutions with the best known solutions (BKSs). The first 23 test instances, namely p01 through p23 have maximum tour length constraints, i.e. customer service times are ignored. The remaining 10 instances (pr01 to pr10) have MTD constraints with a different service time for each customer. The objective values obtained by TSH on all 33 MDVRP instances are provided in Table 3. The left-most column indicates the problem name. The remaining columns are: n (number of customers); m (number of depots), type (the type of instance where C stands for capacity constraint and D for tour duration constraint), BKS (best known solution's objective value), TSH (our algorithm's objective value), PD (percent deviation of TSH from BKS), and the last column shows the CPU time of the TSH. The last row shows average percent deviation and CPU time.

Table 3
Performance of TSH on 33 MDVRP instances

Problem Name	n	m	Type	BKS	TSH	PD (%)	Time (s)
p01	50	4	C	576.87	576.87	0.00	9.4
p02	50	4	C	473.53	473.53	0.00	20.9
p03	75	5	C	641.19	641.19	0.00	141.9
p04	100	2	C	1001.59	1008.47	0.69	258.8
p05	100	2	C	750.03	758.87	1.18	159.1
p06	100	3	C	876.50	881.76	0.59	194.8
p07	100	4	C	885.80	896.96	1.25	94.8
p08	249	2	C,D	4420.95	4430.36	0.21	616.4
p09	249	3	C,D	3900.22	3971.59	1.83	606.0
p10	249	4	C,D	3663.02	3779.10	3.17	703.7
p11	249	5	C,D	3554.18	3652.01	2.75	660.3
p12	80	2	C	1318.95	1318.95	0.00	13.8
p13	80	2	C,D	1318.95	1318.95	0.00	6.7
p14	80	2	C,D	1360.12	1365.69	0.41	34.5
p15	160	4	C	2505.42	2552.79	1.89	255.4
p16	160	4	C,D	2572.23	2572.23	0.00	244.7
p17	160	4	C,D	2709.09	2731.37	0.82	125.2
p18	240	6	C	3702.85	3802.29	2.69	302.9
p19	240	6	C,D	3827.06	3831.71	0.12	207.9
p20	240	6	C,D	4058.07	4097.06	0.96	217.0
p21	360	9	C	5474.84	5617.53	2.61	1703.0
p22	360	9	C,D	5702.16	5706.81	0.08	424.0
p23	360	9	C,D	6095.46	6145.58	0.82	609.2
pr01	48	4	C,D	861.32	861.32	0.00	2.1
pr02	96	4	C,D	1288.37	1303.13	1.15	116.4
pr03	144	4	C,D	1782.58	1810.19	1.55	527.7
pr04	192	4	C,D	2072.52	2130.96	2.82	225.5
pr05	240	4	C,D	2343.66	2473.46	5.54	1206.4
pr06	288	4	C,D	2675.16	2709.08	1.27	1651.2
pr07	72	6	C,D	1085.61	1085.61	0.00	12.8
pr08	144	6	C,D	1666.60	1684.04	1.04	414.8
pr09	216	6	C,D	2153.10	2177.20	1.12	613.5
pr10	288	6	C,D	2811.49	2828.26	0.60	5271.4
<i>Average</i>				2428.17	2460.45	1.13	534.9

The average percent deviation of 1.13% on 33 MDVRP instances is promising. In eight instances PD is equal to 0% (rows in bold). In order to assess the performance of TSH on SMDVRP instances, we

selected two MDVRP instances. The first instance is pr01 where $n = 48$ and $m = 4$. We considered all subsets of m with one or more depots, i.e. $m_{eff} \in \{1, 2, 3, 4\}$ with outsourcing cost of 1.50. So we created $2^m - \binom{m}{0} = 16 - 1 = 15$ distinct SMDVRP instances. The second instance is p03 where $n = 75$ and $m = 5$. For this instance we considered subsets of m with two or more depots, i.e. $m_{eff} \in \{2, 3, 4, 5\}$ with outsourcing cost of 0.80. In this case we created $2^m - \binom{m}{0} - \binom{m}{1} = 32 - 1 - 5 = 26$ distinct SMDVRP instances. Because there exists no best known solution for these 41 SMDVRP instances, we compare the TSH result with the respective solution obtained by CPLEX subject to a time limit of three hours.

Table 4
Performance of CPLEX and TSH on 41 SMDVRP instances

Problem Name	n	m	m_{eff}	ID of the effective depots	$CPLEX$ (3h) UB	$CPLEX$ Gap (%)	TSH	TSH $time$ (s)	$CPLEX$ vs TSH Gap (%)
pr01	48	4	1	1	753.59	0.00	756.43	38.9	0.38
				2	794.78	0.00	800.60	79.2	0.73
				3	768.89	0.00	804.68	95.8	4.66
				4	752.30	0.00	778.56	47.0	3.49
			2	1,2	742.78	0.00	744.37	33.7	0.21
				1,3	701.02	0.00	701.02	75.3	0.00
				1,4	723.05	0.00	728.92	37.9	0.81
				2,3	732.56	0.00	743.79	58.8	1.53
				2,4	725.94	0.00	735.78	58.5	1.36
				3,4	719.18	0.00	727.69	86.3	1.18
			3	1,2,3	698.81	0.00	698.81	44.7	0.00
				1,2,4	717.22	0.00	717.22	37.0	0.00
				1,3,4	693.06	0.00	699.29	41.0	0.90
				2,3,4	695.94	0.00	697.64	46.4	0.24
			4	1,2,3,4	687.23	0.00	688.60	33.1	0.20
p03	75	5	2	1,2	737.24	7.91	745.42	125.6	1.11
				1,3	726.76	6.07	756.95	73.2	4.15
				1,4	737.56	5.44	762.99	117.0	3.45
				1,5	741.05	7.60	760.53	110.6	2.63
				2,3	744.58	7.95	769.13	76.3	3.30
				2,4	725.50	6.69	746.61	90.2	2.91
				2,5	789.39	7.83	812.35	82.0	2.91
				3,4	739.38	6.89	771.19	87.2	4.30
				3,5	737.79	7.23	786.51	168.1	6.60
				4,5	793.06	6.91	827.69	80.2	4.37
			3	1,2,3	682.34	7.14	701.59	89.0	2.82
				1,2,4	701.00	9.03	712.11	54.1	1.58
				1,2,5	694.80	7.83	717.85	123.1	3.32
				1,3,4	691.89	6.80	711.70	50.7	2.86
				1,3,5	689.38	9.43	706.23	53.0	2.44
				1,4,5	724.28	9.80	736.42	80.8	1.68
				2,3,4	658.43	6.95	676.62	52.4	2.76
				2,3,5	678.64	7.86	705.37	90.7	3.94
				2,4,5	703.70	7.90	717.92	55.4	2.02
				3,4,5	686.65	7.74	710.17	96.9	3.43
			4	1,2,3,4	650.09	7.19	653.96	129.7	0.60
				1,2,3,5	650.78	8.25	651.03	88.9	0.04
				1,2,4,5	680.49	9.51	687.99	57.8	1.10
				1,3,4,5	670.47	9.17	682.50	56.9	1.79
				2,3,4,5	654.19	9.39	635.63	53.5	-2.84
			5	1,2,3,4,5	626.58	6.77	631.19	90.9	0.74
Average					712.74	4.91	726.85	74.3	1.94

Results of CPLEX and TSH on 41 SMDVRP instances are presented in Table 4 where m_{eff} stands for the number of operational depots. *CPLEX (3h) UB* and *CPLEX Gap (%)* stand for the best feasible solution (upper bound *UB*) obtained by CPLEX subject to three hours of CPU time limit and for the optimality gap measured as the percent difference between the lower and upper bounds, respectively. The column header *TSH* stands for the best feasible solution obtained by our Tabu Search heuristic. The last column shows the relative gap between *CPLEX (3h) UB* and TSH calculated as $100 \times \frac{TSH - CPLEX(3h) UB}{CPLEX(3h) UB}$. All 15 instances of pr01 were solved to proven optimality by CPLEX within three hours. TSH produces an average deviation of 1.05% from the optimal solution on these instances. The overall performance of TSH is promising also on the 41 SMDVRP instances; average CPU time is 74.33 seconds and average gap is 1.94%.

Table 5
Interdiction patterns of problem p01 ($n=50$ and $m=4$) for $r=1$

Problem Name	ID of the Interdicted Depot	Value	No. Outsourced Customers
p01	1	583.46	11
	2	599.76	11
	3	576.80	8
	4	583.73	19

(a)

(b)

Fig. 1. (a)–(d) Visual representation of obtained solutions for each interdiction pattern in $p01$ with $r = 1$. Fig. (b) shows the best combination p^{best} (best interdiction pattern) obtained by interdicting Depot 2. The interdicted depot and outsourced customers are colored light and dark, respectively.

4.2. Results obtained on RI-SMDVRP

We constructed RI-SMDVRP instances from 33 MDVRP test instances. As an example, consider test instance $p01$ where the number of the depots is equal to four. We set the outsourcing cost (c^o) and the number of depots to be interdicted (r) to one. From the attacker's view point, there are $\binom{4}{1} = 4$ interdiction patterns and for each pattern the defender will optimize the vehicle routes. The pattern which causes the maximum disruption in the network (highest objective value) is the attacker's final decision. Table 5 shows objective function values of the attacker's problem for each pattern, where the maximum disruption is attained by interdicting the **second** depot and outsourcing **11** customers. Fig. 1. presents a visualization of this problem's topology where circles represent customer nodes and squares represent depots.

Table 6
Outsourcing cost for each SMDVRP instance when $r = 0$

Problem Name	c^o	Ratio (%)	Problem Name	c^o	Ratio (%)	Problem Name	c^o	Ratio (%)
p01	1.0	10.00	p12	12.0	10.00	p23	15	10.00
p02	1.0	14.00	p13	10.0	17.50	pr01	1.5	20.83
p03	0.8	10.67	p14	10.0	30.00	pr02	1.5	22.92
p04	1.0	17.00	p15	14.0	10.00	pr03	1.5	20.14
p05	1.0	9.00	p16	15.0	4.38	pr04	1.0	19.79
p06	1.0	12.00	p17	15.0	10.00	pr05	2.0	5.42
p07	1.0	10.00	p18	6.0	30.00	pr06	2.0	7.99
p08	0.7	10.04	p19	15.0	2.50	pr07	2.0	18.06
p09	0.5	8.43	p20	15.0	10.00	pr08	1.5	9.03
p10	0.5	12.05	p21	14.0	10.00	pr09	1.0	26.85
p11	0.5	10.04	p22	15.0	2.22	pr10	2.0	4.51

All three methods were applied on 33 RI-SMDVRP instances which differ in the number of customers and depots. The parameter r is set to 1 (attacker interdicts one depot) and the outsourcing cost for each instance is given in Table 6. The 3rd, 6th and 9th columns of this table show for each RI-SMDVRP instance the ratio of the number of customers outsourced in the best TSH solution when $r = 0$ (no-interdiction case) to the original total number of customers in the instance. We tried to adjust the unit outsourcing cost for each instance in such a way that the solution does not suggest either outsourcing all customers (caused by too a low unit outsourcing cost) or outsourcing no customers at all (caused by too a high unit outsourcing cost).

Table 7
Results of implementing three methods on 33 RI-SMDVRP instances with $r = 1$

Problem Name	n	m	METHOD 1'			METHOD 2				METHOD 3				
			ID	ObjVal	Time (s)	ID	ObjVal	Gap (%)	Time (s)	ID	ObjVal	Gap (%)	Iter.	Time (s)
p01	50	4	2	599.76	410.2	2	577.10	5.20	11372.4	2	577.10	5.20	3	33012.3
p02	50	4	2	467.13	210.6	2	460.81	3.79	11033.8	1	463.98	4.68	4	43491.5
p03	75	5	3	687.99	788.5	3	672.44	7.87	11007.5	3	672.44	7.87	2	21808.6
p04	100	2	1	1187.83	108.1	1	1136.72	4.26	11046.3	1	1136.72	4.26	2	21896.0
p05	100	2	1	888.90	164.4	1	859.34	7.50	11006.4	1	859.34	7.50	1	11006.4
p06	100	3	1	1032.18	132.9	1	1015.52	8.39	11118.7	1	1015.52	8.39	2	21920.2
p07	100	4	4	956.24	294.3	4	913.52	9.83	11169.1	1	929.68	11.02	4	43572.5
p08	249	2	1	5651.51	406.6	1	5651.51	11.45	11644.0	1	5651.51	11.45	1	11644.0
p09	249	3	1	4741.07	250.4	1	4741.07	12.83	11931.4	1	4741.07	12.83	1	11931.4
p10	249	4	2	4238.00	246.5	2	4238.00	16.17	11836.2	2	4238.00	16.17	1	11836.2
p11	249	5	2	4004.66	254.0	2	4004.66	18.03	12239.7	2	4004.66	18.03	1	12239.7
p12	80	2	1	1802.33	72.6	1	1767.16	7.36	10896.8	1	1767.16	7.36	2	21714.2
p13	80	2	1	2678.72	192.9	1	2660.91	39.54	11014.1	1	2660.91	39.54	2	21832.7
p14	80	2	1	2778.72	165.8	1	2764.57	41.89	10977.0	1	2764.57	41.89	2	21787.3
p15	160	4	3	3079.35	333.8	3	2968.15	15.96	11144.2	2	3026.03	17.36	3	32746.6
p16	160	4	1	4785.16	519.1	1	4697.25	46.56	11334.9	1	4697.25	46.56	4	43742.5
p17	160	4	1	5174.04	228.2	1	5047.78	50.32	11330.8	1	5047.78	50.32	4	43736.2
p18	240	6	6	3850.64	680.5	6	3791.23	11.75	11859.2	2	3801.84	11.96	4	44268.1
p19	240	6	6	6061.62	541.8	6	4926.82	28.04	12433.4	4	5585.32	38.77	6	66438.3
p20	240	6	6	6476.44	307.2	6	5064.01	30.11	12582.3	1	5810.38	41.07	6	66688.5
p21	360	9	7	6284.56	3127.3	7	6284.56	-	16116.4	7	6284.56	-	1	16116.4
p22	360	9	5	7946.25	634.9	5	7946.25	-	16643.1	5	7946.25	-	1	16643.1
p23	360	9	4	8405.54	873.3	4	8405.54	-	18571.1	4	8405.54	-	1	18571.1
pr01	48	4	3	717.22	170.6	3	717.22	0.00	1331.3	3	717.22	0.00	1	1331.3
pr02	96	4	1	1009.87	136.3	1	1006.59	8.92	11042.4	1	1006.59	8.92	1	11042.4
pr03	144	4	2	1772.16	51.7	2	1659.07	14.16	10974.8	2	1659.07	14.16	3	32576.8
pr04	192	4	4	1883.28	87.9	4	1837.56	13.60	11201.6	4	1837.56	13.60	1	11201.6
pr05	240	4	1	2600.29	839.8	1	2600.29	18.19	12668.3	1	2600.29	18.19	1	12668.3
pr06	288	4	3	3003.08	510.2	3	3003.08	16.07	12352.1	3	3003.08	16.07	1	12352.1
pr07	72	6	4	1008.48	503.4	4	960.35	3.97	10951.9	5	976.61	6.19	5	54154.7
pr08	144	6	5	1673.58	49.0	5	1582.66	11.10	11112.5	6	1626.09	16.09	5	54318.1
pr09	216	6	3	1905.66	625.0	3	1905.66	15.00	11754.6	3	1905.66	15.00	1	11754.6
pr10	288	6	4	2942.90	981.6	4	2942.90	16.58	13230.8	4	2942.90	16.58	1	13230.8
Average					451.5			16.48	11372.4			17.57		26462.9

The results of the three methods applied to 33 RI-SMDVRP instances all with $r = 1$ are given in Table 7. For each method and each instance, the solution value ($ObjVal$), relative CPLEX gap for the best interdiction pattern p^{best} and the solution time are reported. Solution times of METHOD 2 and METHOD 3 are inclusive of those of METHOD 1'. The ID columns indicate the IDs of the interdicted depot obtained by each method. The $Iter.$ column under METHOD 3 shows the number of times CPLEX is called by METHOD 3. From Table 7 we observe that the best interdiction pattern for each instance is obtained by METHOD 3. The criteria for selecting the best interdiction pattern are as follows.

- 1st Criterion:* If all methods find the same ID (the same interdiction pattern), then the best routing plan ro^{best} is the one that is associated with the minimum of the three objective values.
- 2nd Criterion:* If each method finds a different ID (a different interdiction pattern each), then the best routing plan ro^{best} is the one that is associated with the maximum of the three objective values.
- 3rd Criterion:* Suppose the three methods find two identical IDs $j_1 = j_2$ and one different ID j_3 with the respective objective values $ObjVal_1$, $ObjVal_2$ and $ObjVal_3$. We select as the best interdiction pattern $j^* = \arg \max_{j_1, j_2, j_3} (\min(ObjVal_1, ObjVal_2), ObjVal_3)$. The routing plan associated with that pattern becomes ro^{best} where ties are broken arbitrarily.

In 25 RI-SMDVRP instances both METHODS 2 and 3 produce the same results while in 12 large instances all three methods produce the same result. In other words, in large size instances METHODS 2 and 3 cannot produce a better solution than METHOD 1' which implies that METHOD 1' has the potential to tackle problems with a larger number of depots. In fact, in three instances (p21, p22 and p23) CPLEX could not find a lower bound in three hours. The optimal solution (optimal interdiction pattern) for pr01 is obtained by all three methods. The average row suggests that METHOD 1' is the fastest method (451.50 s) and METHOD 3 is the slowest method (26,462.86 s). Table 8 shows the results of three methods on a particular RI-SMDVRP instance, namely pr08 for different values of the interdiction cardinality $r = 2, 3, 4, 5$. In this case, for all values of r , METHODS 2 and 3 produce the same results (same CPU time and same gap %) whereas for $r = 5$ METHODS 2 and 3 cannot improve the solution found by METHOD 1'. Bolded cells in in Table 7 and Table 8 host the best $ObjVal$.

Table 8

Results of three methods on the RI-SMDVRP instance pr08 ($n = 144$, $m = 6$) for $r = 2, 3, 4, 5$

Problem Name	r	METHOD 1'			METHOD 2				METHOD 3			
		ID	$ObjVal$	Time (s)	ID	$ObjVal$	Gap (%)	Time (s)	ID	$ObjVal$	Gap (%)	Time (s)
pr08	2	3,5	1796.17	52.4	3,5	1767.54	14.27	11593.0	3,5	1767.54	14.27	11593.0
	3	3,5,6	1960.99	58.4	3,5,6	1955.49	15.40	11925.3	3,5,6	1955.49	15.4	11925.3
	4	3,4,5,6	2177.86	69.6	3,4,5,6	2142.44	15.12	11691.2	3,4,5,6	2142.44	15.12	11691.2
	5	1,2,3,4,6	2290.82	82.4	1,2,3,4,6	2290.82	14.88	11209.8	1,2,3,4,6	2290.82	14.88	11209.8
Average				65.7			14.92	11604.8			14.92	11604.8

5. Conclusions and future work

In this paper we model and solve a bilevel optimization problem for the determination of the most critical depots in a multi-depot vehicle routing network. We model the problem as an attacker-defender game (Stackelberg game) from the attacker's viewpoint who aims to inflict the maximum disruption on the network. We refer to this problem as the r -interdiction selective multi-depot vehicle routing problem (RI-SMDVRP). The objective of the defender is to optimize the vehicle routes after the attack and satisfy all customer demand at the minimum possible cost either using the remaining depots or through outsourcing to a 3PL. The attacker chooses r depots to interdict and his/her objective is to maximize the same cost function of the defender. To our knowledge, this is the first time a facility interdiction problem is modeled on a routing network where an adversary's "best, thus most disruptive" action plan is sought after. We propose three solution methods to tackle the RI-SMDVRP where the ULP is solved through exhaustive enumeration and the LLP—which is a selective MDVRP or SMDVRP—is solved by a Tabu Search heuristic. After being convinced by the performance of the proposed heuristic on MDVRP and SMDVRP instances, we performed computational experiments on the bilevel RI-SMDVRP instances. They were built from 33 standard MDVRP instances known in the literature with different c^o (outsourcing cost) and r (number of interdiction) values.

In our future work, we will extend the problem by adding depot protection decisions into our bilevel programming framework, and model the new problem as a tri-level defender-attacker-defender game. In this tri-level formulation, the defender in the upper level decides which depots to protect or fortify in order to minimize the sum of the total traveling cost of vehicles and the outsourcing cost incurred due to unsatisfied demand. In the middle level, the attacker plans his/her worst-case (most destructive) interdiction that would destroy r depots among the unprotected ones so as to inflict maximum disruption in the network. Finally, in the lower level the defender minimizes the same objective function of the attacker by optimizing vehicle routes for the surviving depots and deciding which customers' demand to outsource. As a solution approach, we will focus on a matheuristic method combining exact and heuristic techniques which could discover near-optimal solutions within reasonable time.

Another future research challenge could be to consider alternative objectives for the attacker. A straightforward and sensemaking one is the maximization of the total amount of outsourced demand which ignores any cost in the attacker's problem. Building upon this one, one may write the total distance-weighted cost of outsourced demand as yet another objective for the attacker. In the latter case, the outsourcing cost of a customer to be charged by the 3PL to the defender would be linearly dependent both on the volume of demand as well as on his or her distance to the closest operational depot in the wake of interdiction.

Acknowledgments

The authors would like to acknowledge constructive comments and suggestions from two anonymous referees. Their scrupulous reviews proved rather helpful in improving the presentation of the paper.

References

- Aksen, D. and Aras, N., 2006. Customer selection and profit maximization in vehicle routing problems. In *Operations Research Proceedings 2005* (pp. 37–42). Springer, Berlin, Heidelberg.
- Aksen, D., Piyade, N. and Aras, N., 2010. The budget constrained r -interdiction median problem with capacity expansion. *Central European Journal of Operations Research*, 18(3), pp.269–291.

- Aksen, D., Öncan, T. and Sadati, M.E.H., 2018. An empirical investigation of four well-known polynomial-size VRP formulations. *arXiv preprint arXiv:1810.00199*.
- Aras, N., Aksen, D. and Tekin, M.T., 2011. Selective multi-depot vehicle routing problem with pricing. *Transportation Research Part C: Emerging Technologies*, 19(5), pp.866–884.
- Arroyo, J.M. and Galiana, F.D., 2005. On the solution of the bilevel programming formulation of the terrorist threat problem. *IEEE Transactions on Power Systems*, 20(2), pp.789–797.
- Church, R.L., Scaparra, M.P. and Middleton, R.S., 2004. Identifying critical infrastructure: the median and covering facility interdiction problems. *Annals of the Association of American Geographers*, 94(3), pp.491–502.
- Church, R.L. and Scaparra, M.P., 2007. Protecting critical assets: the r -interdiction median problem with fortification. *Geographical Analysis*, 39(2), pp.129–146.
- Clarke, G. and Wright, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4), pp.568–581.
- Cordeau, J.F., Gendreau, M. and Laporte, G., 1997. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks: An International Journal*, 30(2), pp.105–119.
- Cordeau, J.F. and Maischberger, M., 2012. A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & Operations Research*, 39(9), pp.2033–2050.
- Crevier, B., Cordeau, J.F. and Laporte, G., 2007. The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research*, 176(2), pp.756–773.
- Dantzig, G., Fulkerson, R. and Johnson, S., 1954. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, 2(4), pp.393–410.
- Feillet, D., Dejax, P. and Gendreau, M., 2005. Traveling salesman problems with profits. *Transportation Science*, 39(2), pp.188–205.
- Gavish, B. and Graves, S.C., 1978. The travelling salesman problem and related problems. Working paper GR-078-78, *Operations Research Center*, Massachusetts Institute of Technology, Boston, 1978.
- Gendreau, M., 2003. An introduction to tabu search. In *Handbook of Metaheuristics* (pp. 37–54). Springer, Boston, MA.
- Gendreau, M., Iori, M., Laporte, G. and Martello, S., 2008. A Tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. *Networks: An International Journal*, 51(1), pp.4–18.
- Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5), pp.533–549.
- Israeli, E. and Wood, R.K., 2002. Shortest-path network interdiction. *Networks: An International Journal*, 40(2), pp.97–111.
- Kleinberg, J. and Tardos, E. (2006). *Algorithm design*. Pearson Education India.
- Laporte, G., 1984. Optimal solutions to capacitated multidrop vehicle routing problems. *Congressus Nemerantium*, 4, pp.283–292.
- Laporte, G., Gendreau, M., Potvin, J.Y. and Semet, F., 2000. Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operational Research*, 7(4-5), pp.285–300.
- Lenstra, J.K. and Rinnooy Kan, A.H.G., 1981. Complexity of vehicle routing and scheduling problems. *Networks: An International Journal*, 11(2), pp.221–227.

- Lopes, D., Ramos, T.R.P. and Barbosa-Póvoa, A.P., 2015. Optimization of a recyclable waste collection system- the valorsul case study. In *Operations Research and Big Data* (pp. 97–105). Springer, Cham.
- McMasters, A.W. and Mustin, T.M., 1970. Optimal interdiction of a supply network. *Naval Research Logistics Quarterly*, 17(3), pp.261–268.
- Moore, J.T. and Bard, J.F., 1990. The mixed integer linear bilevel programming problem. *Operations Research*, 38(5), pp.911–921.
- Quintero-Araújo, C.L., Juan, A.A., Montoya-Torres, J.R. and Muñoz-Villamizar, A., 2016, December. A simheuristic algorithm for horizontal cooperation in urban distribution: application to a case study in Colombia. In *Proceedings of the 2016 Winter Simulation Conference* (pp. 2193–2204). IEEE Press.
- Salmeron, J., Wood, K. and Baldick, R., 2004. Analysis of electric grid security under terrorist threat. *IEEE Transactions on Power Systems*, 19(2), 905–912.
- Scaparra, M.P. and Church, R., 2012. Protecting supply systems to mitigate potential disaster: a model to fortify capacitated facilities. *International Regional Science Review*, 35(2), pp.188–210.
- Stenger, A., Vigo, D., Enz, S. and Schwind, M., 2013. An adaptive variable neighborhood search algorithm for a vehicle routing problem arising in small package shipping. *Transportation Science*, 47(1), pp.64–80.
- Stenger, A., Schneider, M. and Goeke, D., 2013. The prize-collecting vehicle routing problem with single and multiple depots and non-linear cost. *EURO Journal on Transportation and Logistics*, 2(1-2), pp.57–87.
- Subramanian, A., Uchoa, E. and Ochi, L.S., 2013. A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, 40(10), pp.2519–2531.
- Talbi, E.-G., 2009. *Metaheuristics: From Design to Implementation* (Vo. 74). John Wiley & Sons.
- Toth, P. and Vigo, D., 2003. The granular tabu search and its application to the vehicle-routing problem. *INFORMS Journal on Computing*, 15(4), pp.333–346.
- Vidal, T., Crainic, T.G., Gendreau, M. and Prins, C., 2014. Implicit depot assignments and rotations in vehicle routing heuristics. *European Journal of Operational Research*, 237(1), pp.15–28.
- Von Stackelberg, H., 1952. *The Theory of the Market Economy*. Oxford University Press.
- Wollmer, R., 1964. Removing arcs from a network. *Operations Research*, 12(6), pp.934–940.
- Wollmer, R.D., 1970. Algorithms for targeting strikes in a lines-of-communication network. *Operations Research*, 18(3), pp.497–515.

Appendix A: Iterative Marginal Cost Analysis

1-Node iMCA

The marginal cost of a given customer i is defined as in Fig. A1 where k and l are the predecessor and successor of customer i , respectively. If i is not worth visiting, then it can be dropped from the route. In other words, customer i should be outsourced. Otherwise, the cost of visiting customer i turns out to be less than its outsourcing cost. Therefore, visiting customer i is desirable, and it should stay between customers k and l . The pseudo code of 1-Node iMCA is provided in Table A1. The pseudo code of 2-Node iMCA can be derived in a straightforward way from there.

Fig. A1. 1-Node Marginal Cost Analysis

2-Node $iMCA$

The marginal cost of a given chain of two customers i and j is defined as in Fig. A2 where k and l are the predecessor and successor of customer i and customer j , respectively. If the chain of customers i and j is not worth visiting, then they can be dropped from the route. In other words, i and j should be outsourced. Otherwise, the cost of visiting this chain turns out to be less than its outsourcing cost. Therefore, visiting customers i and j is desirable, and they should stay as a chain between customers k and l .

Fig. A2. 2-Node Marginal Cost Analysis

Table A1.

The pseudo code of 1-Node iMCA

Notation	
R :	Current set of routes.
N_r :	Subset of customers on route $r \in R$.
MC_i :	Marginal cost of customer i .
$Index[MC_{[1]}]$:	Index of customer with the highest marginal cost $MC_{[1]}$.
$succ(i), pred(i)$:	Successor and predecessor of customer i , respectively.
1:	For every route $r \in R$
2:	For every customer $i \in N_r$ on route r
3:	Set $MC_i = d_{ki} + d_{il} - d_{kl} - c^o q_i$; // Compute marginal cost of each customer on route $r \in R$.
4:	End For
5:	Sort MC_i values in descending (nonincreasing) order and create a sorted stack S ;
6:	While ($ N_r > 0$)
7:	Retrieve $MC_{[1]} = Pop(S)$; // Return and remove the highest marginal cost.
8:	If $MC_{[1]} < 0$ // Marginal cost of all customers are negative.
9:	Break While loop; // Stop the marginal cost analysis on the current route r .
10:	Else
11:	Set $i_{[1]} = Index[MC_{[1]}]$; // Customer i with highest marginal cost.
12:	Remove $i_{[1]}$ from the route $r \in R$; // Remove the customer with the most positive MC.
13:	Update the MC values of $succ(i_{[1]})$ and $pred(i_{[1]})$;
14:	Restore the nondecreasing order of MC_i in the sorted stack S ;
15:	Update $r \in R$;
16:	Update N_r ;
17:	If $ N_r = 0$ // Route r does not visit any customers.
18:	Discard route $r \in R$;
19:	End If
20:	End If
21:	End While
22:	End For

Appendix B: Tabu search heuristic (TSH)

Table B1

The pseudo code of TSH for the LLP (SMDVRP)

Notation	
$\mathcal{N}$:	Set of customers.
R_{S_t} :	Set of all routes of solution S_t .
$Cost[r]$:	Cost of route $r \in R_{S_t}$.
C^o :	Outsourcing cost.
$totLoad[r]$:	Total load of route $r \in R_{S_t}$.
d_{ij} :	Distance from node i to node j .

S_t :	Solution at iteration t .
$Z(S_t)$:	Cost (objective value) of solution S_t .
S^* :	Best known feasible solution found so far (incumbent).
$Z(S^*)$:	Cost (objective value) of incumbent S^* .
$N(S_t)$:	Neighborhood of solution S_t .
$N_{sub}(S_t)$:	Subset of $N(S_t)$
n_{S_t} :	Number of customers in solution S_t .
v_{S_t} :	Number of routes in solution S_t .
α, β :	Penalty parameters for capacity and time-infeasibility.
δ :	Parameter to update α, β .
λ :	Tabu duration.
$\theta_{i_{N(S_t)}}$:	Last iteration for which customer $i \in \mathcal{N}$ is declared tabu by neighborhood $N(S_t)$.
$\rho_{i_{N(S_t)}}$:	Number of the times customer $i \in \mathcal{N}$ is moved by neighborhood $N(S_t)$.
ω :	Aspiration criterion.
φ :	Granularity threshold.
$AdmissArc$:	Subset of arcs that are admissible according to the granularity threshold.
τ :	Total number of iterations.
$MaxNonImp$:	Maximum number of non-improving iterations.

// Initialization

- 1: Construct the initial solution S_0 and compute $Z(S_0)$; // Using CWH+iMCA described in Section 3.1.1.
- 2: Set $S^* = S_0$, $Z(S^*) = Z(S_0)$;
- 3: Set $\alpha = \beta = 1$, $\delta = 0.5$, $\lambda = \lceil 7.5 \log_{10} n_{S_0} \rceil$ and $\omega := Z(S_0)$;
- 4: Set $\tau = 15000$ and $MaxNonImp$ as specified in Table 2; // Setting the termination criteria.
- 5: **For** every $i \in \mathcal{N}$ and $N(S_0)$
- 6: Set $\theta_{i_{N(S_0)}} = 0$, $\rho_{i_{N(S_0)}} = 0$;
- 7: **End For**

// Constructing Admissible Arcs (Granular Neighborhood)

- 8: Set $\varphi := \frac{Z_0}{(n_{S_0} + v_{S_0})}$;
- 9: **For** every arc (i, j) in the graph
- 10: **If** $(d_{ij} \leq \varphi)$
- 11: $AdmissArc = AdmissArc \cup (i, j)$; // Add arc (i, j) to the set of admissible arcs.
- 12: **End If**
- 13: **End For**

// Tabu Search Main loop

- 14: Initialize $Counter = 0$;
- 15: **For** $t = 0, \dots, \tau$
- 16: Set $Counter = Counter + 1$;
- 17: Generate $N_{sub}(S_t)$ as a subset of $N(S_t)$ based on $AdmissArc$;
- 18: Compute $Z'(S_t)$ according to Eq. (29);

```

19: For each solution  $S' \in N_{sub}(S_t)$ 
20:   Compute  $Z'(S')$  according to Eq. (29);
21:   If  $Z'(S') \geq Z'(S_t)$ 
22:     Update  $Z'(S') = Z'(S') + \sqrt{n_{S_t} v_{S_t}} \left( \frac{\rho_{i_{N(S_t)}}}{t+1} \right) * (Z'(S') - Z'(S_t));$  // Diversification
23:   End If
24: End For
25: Select the solution  $S' \in N_{sub}(S_t)$  with the minimum  $Z'(S')$ ;
26: If  $(\theta_{i_{N(S_t)}} < t)$  OR  $(S' \text{ is feasible AND } Z'(S') < \omega)$  // Check for Aspiration Criteria.
27:   Update  $S_{t+1} = S'$ ,  $Z(S_{t+1}) = Z(S')$ ;
28:   Update  $\theta_{i_{N(S_t)}} = t + \lambda$ ,  $\rho_{i_{N(S_t)}} = \rho_{i_{N(S_t)}} + 1$ ;
29: End If
30: If  $(Z(S') < Z(S^*))$  AND  $(S' \text{ is feasible})$  // Updating the incumbent if necessary.
31:   Update  $S^* = S'$ ,  $Z(S^*) = Z(S')$ ,  $\omega = Z(S')$ ;
32:   Reset  $Counter = 0$ ;
33:   Apply NSP to  $S^*$  which may yield a better  $Z(S^*)$ ; // Intensification
34: End If
35: If  $S'$  is capacity and time-feasible
36:   Update  $\alpha = \alpha / (1 + \delta)$  and  $\beta = \beta / (1 + \delta)$ ;
37: Else
38:   Update  $\alpha = \alpha(1 + \delta)$  and  $\beta = \beta(1 + \delta)$ ;
39: End If
40: If  $Counter = MaxNonImp$  // Second termination criterion
41:   Go To Step 47;;
42: End If
43: If  $t$  is divisible by  $3n_{S'}$  // Updating the set of admissible arcs.
44:   Reset  $AdmissArc = \emptyset$ ;
45:   Set  $\varphi = \frac{Z(S')}{(n_{S'} + v_{S'})}$ ;
46:   Repeat Steps 8:–12: to populate the set  $AdmissArc$ ;
47: End If
48: End For // Tabu Search Main loop
49: For each route  $r \in R_{S^*}$  // Performing Step 5: in Table 1
50:   If  $Cost[r] > (C^0 \times totLoad[r])$ 
51:     Discard  $r$  from  $S^*$  with all its assigned customers;
52:     Set  $S^* = S^* \setminus r$ ;
53:     Set  $Z(S^*) = Z(S^*) - Cost[r] + (C^0 \times totLoad[r])$ ;
54:   End If
55: End For
56: Return  $S^*$  as the feasible best SMDVRP solution and  $Z(S^*)$  as its objective value;

```