Computers and Operations Research 123 (2020) 104996

Contents lists available at ScienceDirect

Computers and Operations Research

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

A trilevel *r*-interdiction selective multi-depot vehicle routing problem with depot protection

Mir Ehsan Hesam Sadati [a], Deniz Aksen [b,*], Necati Aras [c]

[a] Faculty of Engineering and Natural Sciences, Sabancı University, İstanbul, Turkey
[b] College of Administrative Sciences and Economics, Koç University, İstanbul, Turkey
[c] Department of Industrial Engineering, Boğaziçi University, İstanbul, Turkey

ARTICLE INFO

Article history:
Received 10 August 2019
Revised 31 March 2020
Accepted 6 May 2020
Available online 25 May 2020

Keywords:
Trilevel optimization
Protection
Interdiction
Outsourcing
Selective multi-depot vehicle routing problem
Variable neighborhood descent
Tabu search

ABSTRACT

The determination of critical facilities in supply chain networks has been attracting the interest of the Operations Research community. Critical facilities refer to structures including bridges, railways, train/metro stations, medical facilities, roads, warehouses, and power stations among others, which are vital to the functioning of the network. In this study we address a trilevel optimization problem for the protection of depots of utmost importance in a routing network against an intelligent adversary. We formulate the problem as a defender-attacker-defender game and refer to it as the trilevel *r*-interdiction selective multi-depot vehicle routing problem (3LRI-SMDVRP). The defender is the decision maker in the upper level problem (ULP) who picks u depots to protect among m existing ones. In the middle level problem (MLP), the attacker destroys r depots among the $(m–u)$ unprotected ones to bring about the biggest disruption. Finally, in the lower level problem (LLP), the decision maker is again the defender who optimizes the vehicle routes and thereby selects which customers to visit and serve in the wake of the attack. All three levels have an identical objective function which is comprised of three components. (i) Operating or acquisition cost of the vehicles. (ii) Traveling cost incurred by the vehicles. (iii) Outsourcing cost due to unvisited customers. The defender aspires to minimize this objective function while the attacker tries to maximize it. As a solution approach to this trilevel discrete optimization problem, we resort to a smart exhaustive enumeration in the ULP and MLP. For the LLP we design a metaheuristic algorithm that hybridizes Variable Neighborhood Descent and Tabu Search techniques adapted to the Selective MDVRP (SMDVRP). The performance of this algorithm is demonstrated on 33 MDVRP benchmark instances existing in the literature and 41 SMDVRP instances generated from them. Numerical experiments on a large number of 3LRI-SMDVRP instances attest that our comprehensive method is effective in dealing with the defender-attacker-defender game on multi-depot routing networks.

© 2020 Elsevier Ltd. All rights reserved.

1. Introduction and literature review

There has been an increasing interest in the protection and interdiction of critical structures such as bridges, railways, medical facilities, roads, warehouses, train/metro stations, police and power stations among others. This has given rise to the so-called interdiction models. The first interdiction problem was studied by Wollmer (1964). In this seminal study the author considered the interdiction of a fixed number of capacitated arcs in a network so as to reduce the capacity of the network as much as possible. In another study Israeli and Wood (2002) addressed the shortest path problem and determined the critical arcs in it. Other network interdiction models developed in the literature are due to McMasters and Mustin (1970), Salmeron et al. (2004), and Arroyo and Galiana (2005). The study of Church et al. (2004) is the first published work about interdicting nodes in supply chain networks. The authors considered two problems called the *r*-interdiction median problem (RIM) and the *r*-interdiction covering problem (RIC). The aim in these problems is finding a subset of r facilities among p existing ones the interdiction of which yields the maximum disruption in a median and coverage type network, respectively.

Interdiction models are typically nested in a bilevel optimization structure, which is known as the *Stackelberg* or *Leader-Follower Game* (Von Stackelberg, 1952). The decision maker of the upper level problem (ULP) is the attacker, while that of the

* Corresponding author.
E-mail address: daksen@ku.edu.tr (D. Aksen).

https://doi.org/10.1016/j.cor.2020.104996
0305-0548/© 2020 Elsevier Ltd. All rights reserved.

lower level problem (LLP) is the defender. The interdiction model as such becomes an *attacker-defender* game. In other interdiction models such as the one proposed in Church and Scaparra (2007), the goal is to establish the best protection policy of the defender against the disruptive attacks of its opponent. This corresponds to a bilevel *defender-attacker* game where the roles of the attacker and the defender are switched. In Church and Scaparra (2007) the protection decision is embedded in the RIM model of Church et al. (2004), and the resulting model in named as the *r*-interdiction median problem with fortification (RIMF). The objective of this problem is to obtain a subset of facilities that have to be fortified so that the total cost of satisfying the customer demands from operational facilities after the attack is minimized. Aksen et al. (2013) analyzed another bilevel *p*-median problem where the defender in the ULP identifies the facilities to be opened and those to be protected simultaneously. In response the attacker decides in the LLP the facilities to destroy. Sadati et al. (2020) proposed the *r*-interdiction selective multi-depot vehicle routing problem (RI-SMDVRP) which is modelled as an attacker-defender game from the viewpoint of an intelligent adversary whose aim is to cause the highest damage on a routing network. The adversary is the decision maker in the ULP. He determines r out of m existing depots to attack, while the defender in the LLP finds the best vehicle tours after the attack. Other bilevel interdiction models can be found in Brown et al. (2005), Scaparra and Church (2008), Aksen et al. (2010), Aksen and Aras (2012), Aksen et al. (2014) and Aliakbarian et al. (2015).

The study of interdiction in routing networks is a comparatively unexplored venue in the literature confined to the interdiction of arcs within the framework of a Stackelberg game. We are aware of only four papers. Lozano et al. (2017) solve a problem in the context of the traveling salesman problem (TSP) using a backward sampling framework-based exact approach tailored to the bilevel TSP. In their problem the defender decides on a subset of arcs to protect in the first level, the attacker attacks a subset of unprotected arcs (thus increasing their costs) in the second level, and the defender eventually determines the best TSP tour in the network. The other three papers are due to Kheirkhah et al. (2016a, b) and Bidgoli and Kheirkhah (2018). They all model the phenomenon of budget-constrained arc interdiction as an attacker-defender game where the attacker interdicts the arcs of a routing network to maximize the defender's routing cost on the surviving network. The success probability of interdiction of a targeted arc is one. Each arc has its own interdiction cost. The attacker has a limited budget which prevents him from interdicting all arcs. The defender must deliver all customer demands after the attack. The network graph under consideration is complete, and there are arcs between all pairs of customer nodes. There exists only one depot with impunity against destruction from where vehicles are dispatched to visit customers.

We are aware of only five published works in the context of trilevel interdiction problems. Yao et al. (2007) address a trilevel optimization problem to allocate resources in defending an electric power network. The aim is to determine the most critical components to protect against potential terrorist attacks. The problem is modeled as a defender-attacker-defender game where the defender's objective is to minimize the economic cost. The authors develop an exact solution technique based on decomposition which consists of iteratively solving smaller nested bilevel problems.

San Martin (2007) proposes and solves in his Master's thesis a trilevel programming model to generate an optimal plan to defend an infrastructure from a malicious man-made attack. In his model, the defender uses restricted defensive resources for the protection of system components. Then, the attacker deploys finite offensive resources to attack components that are not protected. Finally, the defender guides system operations with an optimization model. The author suggests four decomposition algorithms: direct, nested, reformulation-based, and reordering-based to solve the model.

The RIMF model in Church and Scaparra (2007) is extended in Scaparra and Church (2012) by taking into account capacitated facilities that are vulnerable to attack. The new problem is investigated by means of a trilevel optimization model, where the defender (system planner) considers the protection of certain facilities in the ULP with the goal of minimizing the total cost caused by an attack. In the middle level problem (MLP), r facilities are interdicted by the attacker to inflict the highest damage on the network. Lastly, the defender minimizes in the LLP the demand-weighted total traveling cost by rerouting demands to the operational facilities. They proposed a solution approach consisting of a tree search strategy and showed how the MLP and LLP can be reduced to a single-level problem.

Mahmoodjanloo et al. (2016) address a trilevel facility defense location model for the full coverage in RIM. Their trilevel model is based on the defender-attacker-defender framework. The aim of the proposed model is to construct an appropriate service system so that its full capacity is utilized in order to provide the services during the post-attack period. As a solution approach they proposed three methods which consist of explicit enumeration, genetic and biogeography-based algorithms.

Akbari-Jafarabadi et al. (2017) investigate a trilevel RIM with the goal of minimizing the sum of the pre-interdiction and post-interdiction costs. The attacker disrupts the system under uncertainty where facility opening costs and customer demands are several of the uncertain parameters. The authors develop four hybrid meta-heuristics based on Tabu Search, Rainfall Optimization, and Random Greedy Search.

We introduce in this paper a novel trilevel optimization problem to identify the depots of utmost importance in a multi-depot vehicle routing problem which is modelled as a defender-attacker-defender game from the viewpoint of a defender who intends to protect a fixed number of depots to minimize the total cost after the interdiction of several unprotected depots by the attacker. We call this problem the trilevel *r*-interdiction selective multi-depot vehicle routing problem (3LRI-SMDVRP). The defender in the ULP makes a decision on the depots to be protected. In the MLP, the attacker identifies which r depots to strike among the unprotected ones so that the defender's total post-attack cost is maximized. Finally, in the LLP, the defender aims to minimize the same total cost which covers also demand outsourcing. To this end, he optimizes the vehicle routes using the non-interdicted (active) depots.

In our trilevel problem, each customer's demand should be satisfied by the defender using one of the following two options: dispatching vehicles from active depots and outsourcing the demand to a third-party logistics service provider (3PL). This implies that the defender must decide on the following: which customers should be visited by the fleet of vehicles of the defender's company and which ones should be outsourced to a 3PL. We account for the outsourcing option of the defender by incorporating the cost of outsourcing into his objective function, and refer to the LLP as the Selective Multi-Depot Vehicle Routing Problem (SMDVRP). SMDVRP is an extension of MDVRP which has shown to be *NP*-hard (Lenstra and Rinnooy Kan, 1981). Therefore, finding an optimal solution is difficult with exact approaches even for moderate size instances. As a matter of fact, heuristic techniques have turned out to be a viable method for the solution of SMDVRP. The most relevant papers about SMDVRP are due to Aras et al. (2011), Stenger et al. (2013a) and Stenger et al. (2013b). In this paper,

we propose a hybrid metaheuristic algorithm that combines Variable Neighborhood Descent and Tabu Search heuristics to solve SMDVRP. We then encapsulate this hybrid algorithm in a smart exhaustive enumeration which is complemented by two reduction techniques. This comprehensive methodology constitutes a viable solution approach to the trilevel optimization problem 3LRI-SMDVRP.

A possible application of 3LRI-SMDVRP arises in healthcare management, particularly in drug distribution, where drugs are carried every day from depots to pharmacies and hospitals so as to ensure quick and reliable service. Actually, these depots take the role of transshipment nodes between drug producers and their customers such as pharmacies and hospitals. The disruption in the operations of one or more depots in a metropolitan area would impair the distribution of drugs, which put patients in a cumbersome situation. In the wake of an attack, clients that were previously served by a vehicle dispatched from a destroyed depot should be visited by a vehicle to be departed from an undamaged, thus still operational depot. Another context in which the considered problem can be useful is the military where large firearms such as cannon or rockets have to be shipped from the armories to the combat or training zones. Obviously, if an armory is destroyed by the enemy, the combat capability of the defender decreases severely.

The remainder of this paper is structured as follows. In section 2, we present a mathematical formulation for the 3LRI-SMDVRP. Section 3 elaborates on the solution methodology developed for 3LRI-SMDVRP. Section 4 consists of the results generated on the basis of a great number of experiments. The last section, Section 5, presents concluding remarks and directions for future study.

2. Trilevel formulation of the 3LRI-SMDVRP

The 3LRI-SMDVRP has the same objective function in all levels of the problem, but in opposite directions. In the ULP the defender determines which depots to protect, whereas in the MLP the attacker selects which unprotected depots to destroy. Finally, in the LLP the defender identifies how the demand of each customer must be satisfied as efficiently as possible after the attack by optimizing the vehicle routes. This problem can be modeled by means of a trilevel programming framework in which the defender is the player making the decisions in the ULP and LLP, whereas the attacker is the other player who makes the decision in the MLP. A mixed-integer linear programming (MILP) formulation of the 3LRI-SMDVRP is given below.

Index sets:

$C = \{1, \cdots, n\}$: The set of n customers,
$D = \{n+1, \cdots, n+m\}$: The set of m depots,
$I = C \cup D$: The set of all nodes.

Parameters:

d_{ij} : Traveling cost from node $i \in I$ to node $j \in I$,
c^v : Operating cost per vehicle,
c^o : Unit cost of outsourcing demand,
Q : Vehicle capacity,
q_i : Demand to be collected from customer $i \in C$,
u : Number of depots that can be protected by the defender,
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 $i \in C$,
T_{max}: Maximum tour duration,
M : Big-M value. It can be set to $T_{\max} + \max_{i,j \in C} \{st_j + t_{ij}\}$.

Decision variables:

X_{ij} : 0–1 variable representing whether arc (i,j) is traversed by a vehicle,
V_j : 0–1 variable representing whether the depot at site $j \in D$ is protected,
W_j : 0–1 variable representing whether the depot at site $j \in D$ is interdicted,
$K(\mathbf{V})$: Maximum post-interdiction cost for a given protection strategy $\mathbf{V}$,
$H(\mathbf{W})$: Minimum post-interdiction cost for a given interdiction strategy $\mathbf{W}$,
F_{ij} : Load carried from node $i \in I$ to node $j \in I$ after collecting the demand at i.
Y_{ij} : 0–1 variable representing the assignment of customer $i \in C$ to depot $j \in D$,
A_i : Arrival time at node $i \in I$ where $A_j = 0$ for all $j \in D$ by convention.

3LRI-SMDVRP:

$$\text{Minimize } K(\mathbf{V}) \tag{1}$$

$$\text{subject to} : \sum_{j \in D} V_j = u \tag{2}$$

$$V_j \in \{0,1\} \quad j \in D \tag{3}$$

$$\textit{where } K(\mathbf{V}) = \textit{Maximize } H(\mathbf{W}) \tag{4}$$

$$\textit{subject to} : \sum_{j \in D} W_j = r \tag{5}$$

$$W_j \leqslant 1 - V_j \qquad j \in D \tag{6}$$

$$W_j \in \{0,1\} \qquad j \in D \tag{7}$$

$$\begin{aligned} \text{where } \mathrm{H}(\mathbf{W}) = \textit{Minimize} & \sum_{i \in I} \sum_{\substack{j \in I \\ i \neq j}} d_{ij} X_{ij} + c^v \sum_{j \in D} \sum_{i \in C} X_{ji} \\ & + c^o \sum_{i \in C} q_i \left(1 - \sum_{j \in D} Y_{ij}\right) \end{aligned} \tag{8}$$

$$\text{subject to} : \sum_{i \in C} q_i Y_{ij} \leqslant Q \sum_{i \in C} X_{ji} \tag{9}$$

$$\sum_{i \in D} \sum_{j \in D} X_{ij} = 0 \tag{10}$$

$$\sum_{j \in D} Y_{ij} \leqslant 1 \qquad i \in C \tag{11}$$

$$X_{ij} \leqslant Y_{ij} \qquad i \in C, j \in D \tag{12}$$

$$X_{ji} \leqslant Y_{ij} \qquad i \in C, j \in D \tag{13}$$

$$X_{ik} + Y_{ij} - Y_{kj} \leqslant 1 \qquad i \in C,\ k \in C,\ i \neq k,\ j \in D \tag{14}$$

$$X_{ik} + Y_{kj} - Y_{ij} \leqslant 1 \qquad i \in C,\ k \in C,\ i \neq k,\ j \in D \tag{15}$$

$$\sum_{\substack{j \in I \\ j \neq i}} F_{ij} - \sum_{\substack{j \in I \\ j \neq i}} F_{ji} = q_i \sum_{k \in D} Y_{ik} \qquad i \in C \tag{16}$$

$$F_{ij} \geqslant q_i X_{ij} \qquad i \in I, j \in I, i \neq j \tag{17}$$

$$F_{ij} \leqslant (Q - q_j)X_{ij} \qquad i \in I, j \in I, i \neq j \qquad (18)$$

$$\sum_{\substack{i \in I \\ i \neq j}} X_{ij} - \sum_{\substack{i \in I \\ i \neq j}} X_{ji} = 0 \qquad j \in I \qquad (19)$$

$$X_{ij} + X_{ji} \leqslant 2 - \left\lceil \frac{(q_i + q_j)}{Q} \right\rceil \qquad i \in C, j \in C, ij \qquad (20)$$

$$\sum_{\substack{j \in I \\ i \neq j}} X_{ij} + \sum_{\substack{j \in I \\ i \neq j}} X_{ji} = 2 \sum_{k \in D} Y_{ik} \qquad i \in C \qquad (21)$$

$$Y_{ij} \leqslant 1 - W_j \qquad i \in C,\ j \in D \qquad (22)$$

$$X_{ij} \in \{0, 1\} \qquad i \in I, j \in I \qquad (23)$$

$$Y_{ij} \in \{0, 1\} \qquad i \in C, j \in D \qquad (24)$$

$$F_{ij} \geqslant 0 \qquad i \in I, j \in I \qquad (25)$$

In this formulation, (1)–(3), (4)–(7), and (8)–(25) correspond to the ULP, the MLP, and the LLP, respectively. The defender's objective function is represented by Expression (1) which is comprised of three cost components. (i) Traveling cost incurred by the vehicles. (ii) Operating or acquisition cost of the vehicles. (iii) Outsourcing cost resulting from unsatisfied demand of customers. Constraint (2) guarantees that the defender can only protect u depots. Defender's binary protection variables are defined in (3).

Constraint (5) ensures that the attacker can interdict only r depots. Constraints (6) prevent the interdiction of the depots protected by the defender in the ULP. The binary restriction of the attacker's interdiction variables is ensured in constraints (7). The objective function of the defender in the LLP is given by expression (8) which is the same as the ones in (1) and (4). Constraints (9) put a lower limit on the necessary number of vehicles. The condition that no vehicle can travel between two depots is ensured by constraint (10). Constraints (11) guarantee that at most one depot can be assigned to a customer. Constraints (12) and (13) are logical constraints that make sure that if customer i is not assigned to depot j, then no vehicle can travel from customer i to depot j or vice versa. Constraints (14) and (15) imply that two customers must be assigned to the same depot j if a vehicle visits them consecutively.

Constraints (16) represent the subtour elimination constraints (SECs) adapted from the single-commodity flow formulation of Gavish and Graves (1978) conceived originally for the TSP. In constraints (17) and (18) we enforce tight bounds on the continuous flow variables F_{ij} of that formulation. Note that these bounds were also proposed in Gouveia (1995). Constraints (19) are balance equations, namely the same number of vehicles must enter and leave each depot as well as each customer. Constraints (20) correspond to valid inequalities that help to eliminate subtours of size two. Although they are redundant under the existence of constraints (16)–(18), they help strengthen the SMDVRP formulation in the LLP. Aksen et al. (2018) empirically show that these inequalities provide significant advantage in solving the single-commodity flow formulation of the capacitated vehicle routing problem (CVRP). Constraints (21) are also valid equalities, and make sure that a customer must be visited by a vehicle if it is assigned to a depot. Constraints (11), (19), and (21) together imply that a customer is visited at most once. Constraints link the LLP with the MLP and guarantee that the assignment of customers to an interdicted depot is prohibited. Finally, constraints (23) and (24) define the binary routing and assignment variables of the defender, while constraints (25) are the nonnegativity constraints on the flow variables.

A maximum tour duration constraint (MTD) exists in most of the MDVRP test instances. In order to incorporate this constraint, we define new variables A_i that represent the arrival time at node i and add inequalities (26)–(29) as follows:

$$A_j \geqslant A_i + st_i + t_{ij} - M(1 - X_{ij}) \qquad i \in I, j \in C, i \neq j \qquad (26)$$

$$A_j \leqslant A_i + st_i + t_{ij} + M(1 - X_{ij}) \qquad i \in I, j \in C, i \neq j \qquad (27)$$

$$A_i + st_i + \sum_{j \in D} t_{ij} X_{ij} \leqslant T_{max} \qquad i \in C \qquad (28)$$

$$A_i \geqslant 0 \qquad i \in C \qquad (29)$$

Constraints (26) and (27) together state that if a vehicle travels from node i to node j the arrival time A_j at node j is equal to $A_i + t_{ij} + st_i$. Constraints (28) enforce MTD for each node $i \in C$. Finally, constraints (29) stand for the nonnegativity of the arrival variable A_i.

3. Solution method for the 3LRI-SMDVRP

Moore and Bard (1990) showed that even linear bilevel programming is *NP*-hard. LLP in the 3LRI-SMDVRP is the selective version of the MDVRP, which is known to be *NP*-hard (Lenstra and Rinnooy Kan, 1981). ULP and MLP are binary optimization problems; the bilevel problem comprised of the two is proven to be *NP*-hard (Sadati et al., 2020). A generic instance of this bilevel problem is reducible to a special case of the 3LRI-SMDVRP in which the defender's protective resources are rescinded, i.e. $u = 0$ is the case. From this simple polynomial-time reduction we deduce that the trilevel problem is also *NP*-hard. Hence, a heuristic approach may prove viable for the discovery of an effective solution for it in reasonable time.

The ULP in 3LRI-SMDVRP includes only the protection decision variables that identify the depots to be protected. MLP consists of interdiction decision variables that indicate the unprotected depots to be attacked. We solve ULP and MLP via a smart exhaustive enumeration method that explores a subset of all possible $\binom{m}{u}\binom{m-u}{r}$ combinations of protecting u out of m existing depots and interdicting r among $(m-u)$ unprotected depots assuming that $m \geq u + r$. As mentioned above, the LLP is an SMDVRP where the defender simultaneously optimizes the vehicle routes and makes the outsourcing decisions. For its solution we develop and tailor a hybrid metaheuristic algorithm by combining Variable Neighborhood Descent and Tabu Search Heuristic (VND + TSH).

Note that from the routing problem's point of view there is no difference in the LLP between protected depots and unprotected, but non-interdicted depots; both types of depots remain operational and can dispatch vehicles in the wake of the attack. The defender optimizes the vehicle routes through operational depots only. Therefore, some of the $\binom{m}{u}\binom{m-u}{r}$ protection/interdiction combinations are repeated which include exactly the same interdicted and operational depots, but differ in the identity of the protected depots. To clarify, consider a simple example where the number of depots is equal to $m = 4$, and the cardinality of protected and interdicted depots is equal to 1 $(u = r = 1)$. All $\binom{4}{1}\binom{3}{1} = 12$ protection/interdiction combinations are presented in Table 1. Note that hereafter we use the terms *'pattern'* and *'combination'* interchangeably.

In this table, the indices of the protected and interdicted depots are given for each pattern; the last column shows the active (non-

Table 1
All possible protection and interdiction patterns when $m = 4$ and $u = r = 1$.

Pattern #	Index of the protected depot	Index of the interdicted depot	Indices of operational depots in LLP
1	1	2	1, 3, 4
2	1	3	1, 2, 4
3	1	4	1, 2, 3
4	2	1	2, 3, 4
5	2	3	1, 2, 4
6	2	4	1, 2, 3
7	3	1	2, 3, 4
8	3	2	1, 3, 4
9	3	4	1, 2, 3
10	4	1	2, 3, 4
11	4	2	1, 3, 4
12	4	3	1, 2, 4

interdicted) depots in the LLP. By considering all 12 patterns it can be seen that the final solutions of patterns 1, 8, and 11 lead to the same result. In the first pattern the defender protects the first depot, and in response the attacker interdicts the second depot. Finally, the defender optimizes the problem with the operational depots 1, 3, and 4. In pattern #8, the defender protects the third depot, and in response the attacker interdicts the second depot. Again, in the same way as pattern #1, the defender optimizes the problem by using operational depots 1, 3, and 4. In pattern #11, the defender protects the fourth depot, and in response the attacker interdicts the second depot. The defender optimizes the problem again with the operational depots of 1, 3, and 4. In these three patterns (#1, #8, #11), the index of the interdicted depot is fixed to two, and the indices of the operational depots in LLP are fixed to 1, 3, and 4; they differ only in the indices of the protected depots. This observation holds true for the pattern subsets (#2, #5, #12), (#3, #6, #9), and (#4, #7, #10). So, we need to solve only $\binom{m}{r} = \binom{4}{1} = 4$ combinations in total instead of all 12 combinations. This means that the total number of patterns to be considered depends on m and r, but not on u. In order to avoid solving the same patterns over and over again, we can use a hashing function which will be discussed later in this section. In consequence, we need to solve the LLP in the 3LRI-SMDVRP for $\binom{m}{r}$ distinct patterns instead of all $\binom{m}{u}\binom{m-u}{r}$ possible combinations. We refer to this enumeration scheme as the Smart Exhaustive Enumeration (SEE).

Note that if the underlying MDVRP network of the problem has a symmetric topology comprised of equidistant depots and an equal number of identically clustered customer nodes dispersed around each depot, then the defender's protection problem ULP and the attacker's interdiction problem MLP can have multiple optimal solutions. Due to this perfect symmetry and clustering the optimal objective value $H^*(\mathbf{W})$ of the defender in the LLP would then be the same for a number of distinct protection and associated interdiction strategies ($\mathbf{V}$,$\mathbf{W}$).

To tackle the 3LRI-SMDVRP, we propose two reduction techniques which depend on the cardinality of protected and interdicted depots. The first method is used when $u + r < m$, this is when the sum total of protected and interdicted depots is less than the number of all depots. The second one is employed when$u + r = m$. The aim of these two reduction techniques is to obtain the final solution of 3LRI-SMDVRP by solving the associated LLP for a lesser number of patterns instead of for all $\binom{m}{r}$ patterns.

We refer to the overall solution approach which combines the two reduction techniques and encapsulates the VND + TSH algorithm therein as the Smart Reduced Enumeration (SRE). The reduction techniques in this comprehensive methodology tackle the ULP and MLP of the trilevel problem, while VND + TSH solves the LLP heuristically.

3.1. CASE I ($u + r < m$)

In this case, we solve the related MLP and LLP pair for each protection pattern that is produced during the execution of the SEE method. To this end, we resort to the well-known Parallel Savings Heuristic of Clarke and Wright (1964) and modify it by incorporating with the 1-Node and 2-Node iterative marginal cost analysis (iMCA) explained in the sequel of the paper so that it can be used for solving the SMDVR. We designate the hybrid heuristic as CWH + iMCA, and use it to find the attacker's best interdiction pattern, i.e. the pattern that provides the maximum LLP objective value. For that interdiction pattern, the associated LLP is solved once again with the VND + TSH algorithm by utilizing the CWH + iMCA solution as an initial feasible solution. Then the objective value given by VND + TSH is compared with the one corresponding to the second best interdiction pattern (the pattern for which CWH + iMCA yields the second highest LLP objective value). If the former is larger, we stop because it becomes clear that no interdiction pattern among the remaining ones is able to produce a better (higher) LLP objective value from the perspective of the attacker. Otherwise, for each remaining interdiction pattern for which CWH + iMCA provides a higher LLP objective value than the best objective value of the VND + TSH solution at hand, we solve the associated LLP once again using VND + TSH hoping that a better interdiction pattern for the attacker can be found. At the end of this process, we find the best interdiction pattern for each protection pattern. The final solution of 3LRI-SMDVRP is that particular protection pattern which has the smallest objective function value, thus is the most favorable one from the defender's perspective.

The detailed description of the proposed reduction technique for CASE I (Reduction 1) is given in Algorithm 1. Note that the arrays $Z_{\text{CWH + iMCA}}[]$ and $Z_{\text{VND + TSH}}[]$ store composite records each of which contains a unique pattern pair of p_u and p_r, the associated LLP solution $ro(p_u, p_r)$, and the objective value H of that solution.

Algorithm 1. Reduction 1: Applied when $u + r < m$.

Notation
p_u: A protection pattern given by the SEE method for the ULP.
$PP(p_u)$:Set of protection patterns to be explored in the SEE method.
p_r: An interdiction pattern produced in the SEE method for the solution of the MLP.
$IP(p_r)$:Set of interdiction patterns to be explored in the SEE method.
p_u^{Best}: Best protection pattern.
p_r^{Best}: Best interdiction pattern.
$ro_{\text{CWH + iMCA}}$: LLP routing plan obtained with CWH + iMCA.
$ro_{\text{VND + TSH}}$: LLP routing plan obtained with VND + TSH.
ro^{Best}: Best known routing plan of the 3LRI-SMDVRP.
Z^{Best}: Best known objective value of the 3LRI-SMDVRP.
$Z^{Best}(p_u)$: Best known objective value of the MLP for a given protection pattern p_u.
$Z_{\text{CWH + iMCA}}[]$: The array of LLP solutions obtained by CWH + iMCA.
$Z_{\text{VND + TSH}}[]$: The array of LLP solutions obtained by VND + TSH.
H:The objective value field of each solution record.

(*continued on next page*)

1: Set $Z^{Best} = \infty$;
2: **Loop Over** $PP(p_u)$
3: **Loop Over** $IP(p_r)$ for the current p_u
4: For each p_r solve the ensuing LLP using **CWH + iMCA** and record the solution in $Z_{\text{CWH + iMCA}}[]$;
5: **End Loop**
6: Sort the array $Z_{\text{CWH + iMCA}}[]$[a] in descending (nonincreasing) order of the objective values H;
7: For the particular p_r stored in $Z_{\text{CWH + iMCA}}[1]$, solve the corresponding LLP once again by **VND + TSH** using the routing plan $ro_{\text{CWH + iMCA}}$ in $Z_{\text{CWH + iMCA}}[1]$ as an initial solution. Store the resulting LLP routing plan $ro_{\text{VND + TSH}}$ and its objective value H in $Z_{\text{VND + TSH}}[1]$;
8: Set $Z^{Best}(p_u) = Z_{\text{VND + TSH}}[1].H$, $p_r^{Best} = p_r$, and $ro^{Best} = ro_{\text{VND + TSH}}$;
9: Set $k = 2$;
10: **If** $Z^{Best}(p_u) \geqslant Z_{\text{CWH + iMCA}}[k].H$
11: **Break** and **Go to** Step 24;
// It is clear that no interdiction pattern among the remaining ones [k] through $[\binom{m-u}{r}]$ *provides a higher LLP objective value for the attacker.*
12: **Else**
13: Consider the patterns p_u and p_r and the routing plan $ro_{\text{CWH + iMCA}}$ stored in $Z_{\text{CWH + iMCA}}[k]$. Solve the corresponding LLP once again by **VND + TSH** by using $ro_{\text{CWH + iMCA}}$ as an initial solution, and obtain a new routing plan $ro_{\text{VND + TSH}}$ and a new objective value H to be stored in $Z_{\text{VND + TSH}}[k]$;
14: **If** $Z_{\text{VND + TSH}}[k].H > Z^{Best}(p_u)$
15: Update $Z^{Best}(p_u) = Z_{\text{VND + TSH}}[k].H$, $p_r^{Best} = p_r$, and $ro^{Best} = ro_{\text{VND + TSH}}$;
16: **End If**
17: Set $k = k + 1$;
18: **If** $k > \binom{m-u}{r}$
19: **Break** and **Go to** Step 24;
20: **Else**
21: **Go to** Step 10; *// Move to the p_r with the next largest objective value;*
22: **End If**
23: **End If**
24: **If** $Z^{Best}(p_u) < Z^{Best}$
25: Set $p_u^{Best} = p_u$ and $Z^{Best} = Z^{Best}(p_u)$;
26: **End If**
27: Reset the arrays $Z_{\text{CWH + iMCA}}[]$ and $Z_{\text{VND + TSH}}[]$;
28: **End Loop**
29: Report the incumbent solution record (p_u^{Best}, p_r^{Best}, ro^{Best} and Z^{Best});

[a] $Z_{\text{CWH + iMCA}}[1]$ contains the costliest interdiction pattern, i.e. the best one for the attacker with the highest LLP objective.

As discussed above, thanks to the proposed SEE method, we do not need to explore all $\binom{m}{u}\binom{m-u}{r}$ protection/interdiction patterns. In fact, we revisit several of the previously generated patterns during the implementation of Reduction 1 (Lines 3–5 and Lines 10–23). In order to avoid solving the same LLP more than once, we associate with each pattern a unique decimal integer number called *hash value*. This number represents a pattern as a unique string made of binary digits (bits) 0 and 1. The bits 0 and 1 correspond to a protected/non-interdicted and an interdicted depot, respectively. Let h be the binary string representation of a pattern where the status of the i^{th} depot is shown by bit_i for $i = 1, ..., m$. The unique *hash value* is obtained by the following *hash function:*

$$Hash\ Function(h) = \sum_{i=1}^{m} bit_i \times 2^{m-i} \tag{30}$$

The hash function considerably reduces the computational effort needed to check whether a newly generated protection/interdiction pattern is already stored in the explicit memory. If a pattern is found to be already stored, so must be its associated LLP solution. Thus, the LLP associated with that pattern will not be solved again saving unnecessary runs of the CWH + iMCA and VND + TSH algorithms.

3.2. CASE II ($u + r = m$)

In this case, the defender protects u depots, and in response the attacker interdicts all remaining $m - u$ unprotected depots. In other words, the attacker does not have any decision to make and he/she interdicts all remaining depots and MLP will be voided, i.e., the trilevel problem becomes a one-level problem. When $u + r = m$, we need to explore exactly $\binom{m}{u}\binom{m-u}{r} = \binom{m}{u}\binom{r}{r} = \binom{m}{u} = \binom{m}{m-r} = \binom{m}{r}$ combinations and the final solution of the 3LRI-SMDVRP is the one which has the minimum objective value among all combinations. In lines 10 through 23 of Reduction 1 we try to obtain the best interdiction pattern for a given protection pattern with a lesser number of VND + TSH calls. In CASE II, however, for a given protection pattern there exists only a unique interdiction pattern; therefore, the reduction technique in Reduction 1 cannot be applied to CASE II.

If we use Reduction 1, a separate SMDVRP has to be solved by VND + TSH for each of $\binom{m}{r}$ patterns which certainly is not efficient for large-size instances. We propose a second reduction technique to decrease the solution time. Recall that CWH + iMCA constructs an initial feasible solution for VND + TSH. We surmise that at the end of the VND + TSH implementation the objective value of that initial solution may be improved at most by an average of $\psi \times 100\%$. It leads to the idea of solving only those combinations of which CWH + iMCA solutions can be improved by at least $\psi \times 100\%$ using VND + TSH. Therefore, we first solve the LLP with CWH + iMCA for all protection patterns and obtain the best pattern, i.e. the one that gives the lowest LLP objective value. For the best protection pattern obtained, we solve the associated LLP once again by VND + TSH using the CWH + iMCA solution as the initial feasible solution. The objective value obtained by VND + TSH is then compared to that of the remaining patterns which were obtained by CWH + iMCA. If it is less than $\frac{Z_{\text{CWH + iMCA}}}{1+\psi}$, then we stop; because, according to our surmise it becomes obvious that VND + TSH cannot improve the objective value of the remaining patterns by more than $\psi \times 100\%$. Otherwise, for each remaining pattern whose possible maximum improved objective value is less than the value in the current VND + TSH solution at hand, the corresponding LLP is solved again by VND + TSH to obtain a better pattern.

The detailed description of the proposed reduction technique for CASE II (Reduction 2) is given in Algorithm 2 where we expand the notation in the previous technique for the description of the new technique. We find a proper value for the parameter through a sensitivity analysis which is presented in Section 4.3. The working mechanism of these two reduction techniques is demonstrated by two examples in Appendix A, where the LLP (the pertinent SMDVRP) is solved by using VND + TSH. The proposed VND + TSH algorithm and the generation of an initial solution by CWH + iMCA are elaborated in the next subsection.

Algorithm 2. Reduction 2: Applied when $u + r = m$.

Additional Notation

ψ: Conjectured average percentage by which the objective value H of the CWH + iMCA solution may be improved by VND + TSH. We refer to ψ as the *tolerance parameter* in the sequel.

1: Set a proper value for ψ;
2: **Loop Over** $PP(p_u)$
3: Solve the associated LLP using **CWH + iMCA** and record the solution in $Z_{\text{CWH + iMCA}}[]$;
4: **End Loop**
5: Sort the array $Z_{\text{CWH + iMCA}}[]$[a] in ascending (nondecreasing) order of the objective values H;
6: For the particular p_u stored in $Z_{\text{CWH + iMCA}}[1]$, solve the corresponding LLP once again by **VND + TSH** by using the routing plan $ro_{\text{CWH + iMCA}}$ in $Z_{\text{CWH + iMCA}}[1]$ as an initial solution. Store the resulting LLP routing plan $ro_{\text{VND + TSH}}$ and its objective value H in $Z_{\text{VND + TSH}}[1]$;
7: Set $Z^{Best} = Z_{\text{VND + TSH}}[1].H$, $p_u^{Best} = p_u$, and $ro^{Best} = ro_{\text{VND + TSH}}$;
8: Set $k = 2$;
9: **If** $Z^{Best} \leqslant \frac{Z_{\text{CWH + iMCA}}[k]}{1+\psi}$
10: **Go to** Step 21; *// It can be seen that no remaining pattern [k] through* $[\binom{m}{u}]$ can yield a lower LLP objective for the defender.
11: **Else**
12: Consider the pattern p_u and the routing plan $ro_{\text{CWH + iMCA}}$ stored in $Z_{\text{CWH + iMCA}}[k]$. Solve the corresponding LLP once again by **VND + TSH** using $ro_{\text{CWH + iMCA}}$ as an initial solution, and obtain a new routing plan $ro_{\text{VND + TSH}}$ and a new objective value to be stored in $Z_{\text{VND + TSH}}[k]$;
13: **If** $Z_{\text{VND + TSH}}[k].H \leqslant Z^{Best}$
14: Update $Z^{Best} = Z_{\text{VND + TSH}}[k].H$, $p_u^{Best} = p_u$, and $ro^{Best} = ro_{\text{VND + TSH}}$;
15: **End If**
16: Set $k = k + 1$;
17: **If** $k > \binom{m}{u}$
16: **Go to** Step 21;
18: **Else**
17: **Go to** Step 9; *// Move to the with the next smallest objective value;*
19: **End If**
20: **End If**
21: Report the incumbent solution record (p_u^{Best}, ro^{Best} and Z^{Best});

[a] $Z_{\text{CWH + iMCA}}[1]$ contains the cheapest protection pattern, i.e. the best one for the defender with the lowest LLP objective.

3.3. VND + TSH

VND + TSH can be treated as a hybrid metaheuristic algorithm which combines the Variable Neighborhood Descent method (VND) with the Tabu Search heuristic (TSH) originally devised by Glover (1986). The main constituent of the algorithm is VND which is a deterministic version of the well-known Variable Neighborhood Search (VNS) algorithm invented by Mladenović and Hansen (1997). In VND, the shaking step of VNS that is responsible for the perturbation of the current solution is dismissed, and only the local search step is executed. This local search procedure is implemented efficiently in VND + TSH by applying TSH that makes use of different neighborhood structures which are changed systematically within the VND framework.

The VND + TSH proposed for solving the LLP (SMDVRP) starts with an initial solution S_0 which is generated using the greedy heuristic CWH + iMCA. This step is followed by the definition of a set of move types (neighborhood structures) $N_k (k = 1, ..., k_{\max})$ around $\bar{S} = S_0$. Then a local search is performed by applying the first move type N_1 in a TSH to obtain a new solution S'. The new solution can be either feasible or infeasible. If it is feasible and has a lower objective value than the incumbent solution S^*, S^* is replaced by S' and a new local search step is applied by resetting the move type index k to 1. Otherwise, k is incremented by 1 and the new local search starts from S' using the next neighborhood structure. This process is repeated until all move types are explored. At this point, k is re-initialized and VND + TSH restarts from the solution (feasible or infeasible) obtained in the previous iteration. Finally, these steps are continued until a stopping condition is met—typically the observation of a number of successive non-improving iterations. The pseudo code of the proposed VND + TSH is presented in Algorithm 3.

Algorithm 3. VND + TSH

Notation

C: Set of customers.
R_{S_t}: Set of all routes of solution S_t.
$Cost(r)$: Cost of route $r \in R_{S_t}$.
$TotLoad(r)$: Total load of route $r \in R_{S_t}$.
S_0: Initial solution.
$S_{\text{CWH + iMCA}}$: CWH + iMCA solution.
$\bar{S}$: Current solution.
S^*: Best known feasible solution (incumbent).
$Z(S)$: Cost of solution$S$.
$N_k(S)$: Neighborhood of solution S with respect to move type k.
α, β: Penalty parameters for violation in capacity and maximum tour duration constraints.
λ : Tabu duration.
$\theta_{i_{N_k}}$: The last iteration until the end of which customer $i \in C$ is declared tabu by move type k.
$\rho_{i_{N_k}}$: Number of times customer $i \in C$ is relocated by move type k.
ω: Aspiration criterion to be used in TSH.
AdmissArc: Subset of admissible arcs based on the granularity threshold.
MaxNonImp: Maximum permissible number of successive non-improving iterations.
NSP: Neighborhood search procedure.
t, k: Iteration and move type counters.
`count`:Non-improving iteration counter.

// Initialization

1: Set $S_0 = S_{\text{CWH + iMCA}}$;
2: Set $S^* = S = S_0$ and $Z(S^*) = Z(\bar{S}) = Z(S_0)$;
3: Set $\alpha = \beta = 1$, $\lambda = \lceil 7.5 \log_{10} n_{S_0} \rceil$, $\omega = Z(S^*)$ and *MaxNonImp* as specified in Table 2;
4: Define the set of move types N_k, $k = 1, \ldots, k_{\max}$;
5: Construct *AdmissArc*;
6: **For** every $i \in C$ and $k \in \{1, \ldots, k_{\max}\}$
7: Set $\theta_{i_{N_k}} = 0$ and $\rho_{i_{N_k}} = 0$;
8: **End For**

// VND + TSH main loop begins here.

9: Initialize count = 1 and $t = 1$;

(*continued on next page*)

10: **While** (count $\leqslant$ *MaxNonImp*)
11: Set $k = 1$;
12: **While** ($k \leq k_{\max}$)
13: Set $S' = TSH\left(S,\ k,\ t,\ \theta_{i_{N_k}},\ \rho_{i_{N_k}},\ \omega,\ AdmissArc\right)$; // *Tabu search heuristic as a local search*
14: **If** S' is feasible **AND** $Z(S') < Z(S^*)$
15: Update $S^* = S'$ and $\omega = Z(S^*) = Z(S')$;
16: Reset *count* $= 1$ and $k = 1$;
17: Apply NSP to S^* which may yield a better $Z(S^*)$; // *Intensification*
18: Set $S' = S^*$ if NSP improves $Z(S^*)$;
19: **Else**
20: Set count $=$ count $+ 1$ and $k = k + 1$;
21: **End If**
22: Update *AdmissArc*;
23: Update α and β;
24: Set $t = t + 1$;
25: Set $S = S'$;
26: **End While**
27: **End While**
28: **For** each route $r \in R_{S^*}$
29: **If** $Cost(r) > c^o \times TotLoad(r)$
30: Discard r from S^* with all its assigned customers;
31: Set $S^* = S^*$;
32: Set $Z(S^*) = Z(S^*) - Cost(r) + c^o \times TotLoad(r)$;
33: **End If**
34: **End For**
35: Return S^* as the incumbent SMDVRP solution and $Z(S^*)$ as its objective value;

Generating an initial solution is accomplished by using the method outlined in Sadati et al. (2020). Here, we briefly mention the main steps of the method referred to as CWH + iMCA. The first stage of CWH + iMCA involves solving the given SMDVRP as a traditional MDVRP via the classical Clarke and Wright parallel savings heuristic (CWH) and improving the solution by a rich neighborhood search procedure consisting of the following move types: 1–0 Move, 2–0 Move, 1–1 Exchange, 2–2 Exchange, 2-Opt, 3-Opt, and Or-Opt. It is worth mentioning that because of their complexity 3-Opt and Or-Opt are applied only on single routes as an intra-route operator whereas the other move types are performed within the same route as well as on two different routes. The second stage of CWH + iMCA performs 1-Node and 2-Node iterative marginal cost analysis (iMCA) so as to solve the SMDVRP. The marginal cost of each customer on the basis of routes generated in the first stage is calculated, and subsequently 1-Node and 2-Node iMCA are applied in a row, where single customers and two consecutive customers are either kept or removed based on their marginal costs, respectively. When customers are removed, their demands must be outsourced. This process repeats itself until a negative marginal cost is associated with every customer remaining in the CWH solution. The details of 1-Node and 2-Node iMCA as well as the pseudo code of 1-Node iMCA can be seen in Appendix A of Sadati et al. (2020).

There are 10 move types N_k that are utilized in the local search step of VND + TSH given as follows:

i. Routing move types: 1–0 Move, 1–1 Exchange, 1–2 Move, 2–2 Exchange, 2-Opt.
ii. Customer selection move types: 1-Add, 2-Add, 1-Drop, 2-Drop and 1-Swap.

We remark that the routing move types are applied both within the same route and on two different routes. The first four customer selection move types are explained in detail in Sadati et al. (2020). It suffices to say here that 1-Add inserts a customer not included in a route yet in the best possible position by checking all possible insertion positions. 1-Drop discards a customer currently visited in a route, and connects its predecessor and successor. 2-Add and 2-Drop are the 2-node versions of 1-Add and 1-Drop, respectively. 1-Swap simultaneously adds one customer and drops another one.

From the implementation viewpoint, the move types mentioned above are explored in a cyclic sequential order as proposed by Mladenović and Hansen (1997) starting with N_1 =1–0 Move and ending with N_{10} =1-Swap. The exploration of all aforementioned neighborhood structures in VND + TSH consumes long computation times. In order to speed the neighborhood exploration, we apply granular neighborhood search proposed by Toth and Vigo (2003). The main rationale is the exploration of only a fraction of promising moves by discarding a significant number of unpromising ones. An arc is considered in a move (thus admissible) if its cost (distance) is less than a granularity threshold value defined as $\varphi = \frac{Z}{(n+v)}$ where Z represents the objective value associated with the CWH + iMCA solution, n denotes the number of visited customers, and v is the number of routes in the solution. The admissible arcs are reconstructed anew every $3n$ iterations. This corresponds to a granularity threshold of $\varphi' = \frac{Z'}{(n'+v')}$, where Z' is the objective value obtained by VND + TSH, and n' and v' represent the number of customers and routes in the solution after $3n$ iterations, respectively.

VND + TSH algorithm is supported by strategic oscillation which prescribes to allow infeasible solutions with respect to vehicle capacity and maximum tour duration constraints. Hence, we accept such infeasible solutions by penalizing their objective values using a multiplier proportional to the amount of infeasibility. When this happens, VND + TSH computes the objective value $Z'(S)$ of an infeasible solution S by summing up the original objective value $Z(S)$ with the penalties associated with the capacity infeasibility $Q(S)$ and maximum tour duration infeasibility $D(S)$ as follows:

$$Z'(S) = Z(S) + \alpha Q(S) + \beta D(S) \tag{31}$$

where α and β designate positive penalty parameters corresponding to infeasibilities $Q(S)$ and $D(S)$, respectively. $Z'(S)$ is equal to $Z(S)$ for a feasible solution S, while $Z'(S) > Z(S)$ for an infeasible solution. The values of α and β do not remain constant; they are updated after each iteration so that the best neighboring solution turns out to be feasible or infeasible at approximately the same rate. To this end, we use positive parameter δ and increase the value of α by multiplying it by $(1 + \delta)$ if S is capacity-infeasible. If S is capacity-feasible, we decrease the value of α by dividing it by $(1 + \delta)$. Likewise, if S is time-infeasible (time-feasible), we multiply (divide) the associated penalty parameter β by $(1 + \delta)$. We initialize the values of parameters α and β to one and use the value of 0.5 for δ as proposed in Cordeau et al. (1997).

A crucial element of any tabu search heuristic is the definition of the tabu conditions that are utilized in order to forestall cycling. To this end, tabu conditions associated with each neighborhood structure (move type) are introduced. These conditions prevent each move type from being undone in the next λ iterations where λ denotes the tabu duration used in the TSH. For example, when 1-Add is applied, the customer that is inserted into a given route r cannot be taken off from that route by other move types during the subsequent λ iterations. The tabu condition is revoked for each move either at the end of λ iterations or if it provides a better (smaller) objective value than the incumbent. The latter is called the aspiration criterion. In our TSH implementation, the tabu duration is set to $\lambda = \lceil 7.5 \log_{10} n \rceil$ (Cordeau et al., 1997). We also implement a diversification procedure to explore a larger solution space. In this regard, we apply the diversification strategy that we suc-

cessfully made use of in Sadati et al. (2020). It was suggested by Gendreau et al. (2008), which explores parts of the solution space unexplored previously. Let $\bar{S}$ be the current solution, $\hat{S}$ the new solution obtained by the k^{th} move type, $\rho_{i_{N_k}}$ the number of times the position of customer i is changed by move type k up to now, and t the total number of iterations performed so far. If $Z'(\hat{S}) \geqslant Z'(\bar{S})$, then the penalty term $\eta = \sqrt{n\nu}\left(\frac{\rho_{i_{N_k}}}{t}\right)\left(Z'(\hat{S}) - Z'(\bar{S})\right)$ is added to $Z'(\hat{S})$.

The VND + TSH algorithm is stopped when a predetermined number of successive non-improving iterations is performed. This number is denoted by *MaxNonImp*. We set *MaxNonImp* on the basis of the number of customers in the problem. It also depends on the type of the problem at hand, i.e. whether an MDVRP or an SMDVRP is to be solved. Table 2 shows the specifications of *MaxNonImp* for both problem types. The detailed pseudo code of the proposed TSH is given in Algorithm 4 in Appendix B.

We remark that the TSH subroutine which serves as a local search procedure of the hybrid metaheuristic VND + TSH proposed in this paper has several similarities with the TSH proposed in Sadati et al. (2020). These are: (i) Implementation of a granular neighborhood search structure adopted from Toth and Vigo (2003). (ii) Use of strategic oscillation and the definition of the aspiration criterion therein. (ii) Tabu conditions and the diversification procedure. However, the new TSH differs in the following features:

(i) Application of 1–2 Move and 2–2 Exchange as two additional neighborhood structures (move types) in routing.
(ii) Application of 1-Swap as an additional move type in customer selection.
(iii) The new TSH is applied in the selected neighborhood $N_{sub}(\bar{S})$ by computing the objective values of all neighboring solutions $\hat{S} \in N_{sub}(\bar{S})$ and storing them in an array $Z'(\hat{S})[]$ which is sorted in ascending order. The best one among them which bears the smallest objective value and satisfies at the same time either the tabu conditions or the aspiration criterion (lines 12 through 21 in Algorithm 4 in Appendix B) is selected as the new current solution S'. The TSH proposed in Sadati et al. (2020), on the other hand, selects the solution with the minimum objective value which satisfies either the tabu conditions or the aspiration criterion. If none of these conditions is met, the TSH in Sadati et al. (2020) ignores the current move type, and proceeds with the next one. It does not check the next solution $\hat{S}$ which has the second minimum objective value in the granular neighborhood $N_{sub}(\bar{S})$.
(iv) The new TSH serves as local search phase nested in the main VND routine. Neighborhood structures are explored in a cyclic- sequential order. When a new best solution is found, the local search of the main routine restarts from the first move type of the VND, whereas the TSH of Sadati et al. (2020) is not nested in an upper heuristic.

Table 2
Termination criterion of the VND + TSH algorithm for MDVRP and SMDVRP.

MaxNonImp			
	MDVRP		SMDVRP
$n \leqslant 50$	500	$n \leqslant 50$	500
$50 < n < 100$	150	$50 < n \leqslant 100$	150
$100 \leqslant n < 200$	100	$100 < n$	50
$200 \leqslant n$	50	n/a	n/a

These new features in the proposed TSH alongside its hybridization with VND lead to a significant performance improvement over the previous TSH in Sadati et al. (2020). The results of our numerical experiments which we present in the next section corroborate the superiority of the new TSH.

4. Computational results

4.1. Performance of VND + TSH on MDVRP and SMDVRP instances

In order to assess the performance of VND + TSH on MDVRP and SMDVRP instances, we solved 33 benchmark MDVRP instances and 41 randomly generated SMDVRP instances. Experiments were carried out on a Dell Precision T7810 model computer running on 64-bit Windows 7 with Intel Xeon® E5-2690 v4 2.60 GHz processor and 32 GB ECC RAM. All algorithms were coded in C# of Microsoft® Visual Studio 2017. First, we compared the VND + TSH solutions of 33 MDVRP test instances with the best known solutions in the literature (VRP Web, 2020). We also included in our comparison the results of another tabu search heuristic TSH that was developed earlier in Sadati et al. (2020) to tackle the bilevel version of the *r*-interdiction SMDVRP which was modeled as an attacker-defender game. The employment of the same computing platform enables us to make a fair comparison between the two methods.

The results of this initial comparison can be seen in Table 3. The first column of the table gives the instance name; the second and third columns n and m show the number of customers and the number of depots, respectively; the fourth column Type indicates the type of instance where C stands for capacity constraint and D for maximum tour duration constraint; and the fifth column BKS stands for the best known solution's objective value. These are followed by columns 6, 7 and 8 hosting the Obj. Val. (the objective value returned), the PD (percent deviation from BKS), and the CPU time in seconds for the TSH in Sadati et al. (2020). Finally the last three columns show the same key performance indicators (KPIs) achieved by the new algorithm VND + TSH. Note that PD (%) is calculated as $100 \times \frac{\text{Obj. Val.} - \text{BKS}}{\text{BKS}}$. The bottom row reveals the average KPIs of the two methods. We remark that all tested MDVRP and SMDVRP instances have a symmetric distance matrix which satisfies the triangular inequality.

The average PD of 0.69% on 33 standard MDVRP instances is promising for the new hybrid algorithm VND + TSH, and better than the 1.13% deviation of the TSH in Sadati et al. (2020). In 9 instances PD obtained by VND + TSH is equal to 0% (cells in **bold**), while this number is 8 for TSH. In two instances (p08 and pr07) VND + TSH beats the BKS, and attains negative PD values which are **bolded** as well. Also remarkable is the superiority of VND + TSH over the previous TSH in terms of solution speed. The average solution time of TSH in 33 MDVRP instances is about 535 s, whereas VND + TSH consumes 326 s in average. Note that **bolded** figures in the "Obj. Val." columns of Table 3 indicate the better solution while *italic* numbers in the "Time (s)" columns show the faster solution times for each MDVRP benchmark instance.

To benchmark VND + TSH on SMDVRP instances, we generated 41 instances from two MDVRP instances namely pr01 and p03. In pr01 (n =48, m =4), we consider all subsets of m with one or more depots, i.e. $m_{eff} \in \{1, 2, 3, 4\}$ with an outsourcing cost of $c^o = 1.50$. In p03 (n =75, m =5), we consider subsets of m with two or more depots, i.e. $m_{eff} \in \{2, 3, 4, 5\}$ with $c^o = 0.80$. In total we created 41 distinct SMDVRP instances. Since the BKS for these 41 SMDVRP instances are unknown, we compared the VND + TSH result with the respective solution obtained by CPLEX subject to a time limit of 3 h as well as with the result of the TSH method developed in

Table 3
Performance of VND + TSH on 33 MDVRP instances.

					TSH in Sadati et al. (2020)			The proposed VND + TSH		
Instance	n	m	Type	BKS	Obj. Val.	PD (%)	Time (s)	Obj. Val.	PD (%)	Time (s)
p01	50	4	C	576.87	**576.87**	**0.00**	*9.35*	**576.87**	**0.00**	25.47
p02	50	4	C	473.53	**473.53**	**0.00**	*20.87*	**473.53**	**0.00**	44.80
p03	75	5	C	641.19	**641.19**	**0.00**	141.86	**641.19**	**0.00**	*16.30*
p04	100	2	C	1001.59	1008.47	0.69	258.75	1008.62	0.70	*119.47*
p05	100	2	C	750.03	758.87	1.18	159.10	752.04	0.27	*111.38*
p06	100	3	C	876.50	881.76	0.60	194.81	882.71	0.71	*128.08*
p07	100	4	C	885.80	896.96	1.26	*94.75*	896.01	1.15	95.84
p08	249	2	C,D	4420.95	4430.36	0.21	*616.44*	**4417.34**	**−0.08**	634.39
p09	249	3	C,D	3900.22	3971.59	1.83	*605.99*	3940.56	1.03	751.76
p10	249	4	C,D	3663.02	3779.10	3.17	*703.74*	3696.31	0.91	740.20
p11	249	5	C,D	3554.18	3652.01	2.75	660.30	3578.14	0.67	*506.05*
p12	80	2	C	1318.95	**1318.95**	**0.00**	*13.75*	**1318.95**	**0.00**	15.26
p13	80	2	C,D	1318.95	**1318.95**	**0.00**	*6.74*	**1318.95**	**0.00**	10.89
p14	80	2	C,D	1360.12	1365.69	0.41	34.53	**1360.12**	**0.00**	*23.59*
p15	160	4	C	2505.42	2552.79	1.89	*255.44*	2538.79	1.33	287.63
p16	160	4	C,D	2572.23	**2572.23**	**0.00**	244.70	**2572.23**	**0.00**	*125.92*
p17	160	4	C,D	2709.09	2731.37	0.82	125.20	2731.37	0.82	*95.20*
p18	240	6	C	3702.85	3802.29	2.69	302.91	3798.58	2.59	*195.38*
p19	240	6	C,D	3827.06	3831.71	0.12	207.92	**3827.06**	**0.00**	*66.13*
p20	240	6	C,D	4058.07	4097.06	0.96	217.04	4097.06	0.96	*120.76*
p21	360	9	C	5474.84	5617.53	2.61	*1703.03*	5643.55	3.08	1900.06
p22	360	9	C,D	5702.16	5706.81	0.08	*424.02*	5708.36	0.11	584.47
p23	360	9	C,D	6095.46	6145.58	0.82	609.22	6145.58	0.82	*99.67*
pr01	48	4	C,D	861.32	**861.32**	**0.00**	*2.09*	**861.32**	**0.00**	20.68
pr02	96	4	C,D	1288.37	1303.13	1.15	116.36	1297.70	0.72	*64.94*
pr03	144	4	C,D	1782.58	1810.19	1.55	527.68	1809.14	1.49	*155.33*
pr04	192	4	C,D	2072.52	2130.96	2.82	*225.49*	2088.87	0.79	529.97
pr05	240	4	C,D	2343.66	2473.46	5.54	1206.37	2362.24	0.79	*431.09*
pr06	288	4	C,D	2675.16	2709.08	1.27	1651.23	2695.14	0.75	*773.19*
pr07	72	6	C,D	1085.61	1085.61	**0.00**	*12.75*	**1082.93**	**−0.25**	32.81
pr08	144	6	C,D	1666.60	1684.04	1.05	414.79	1678.73	0.73	*210.40*
pr09	216	6	C,D	2153.10	2177.20	1.12	613.48	2174.63	1.00	*257.36*
pr10	288	6	C,D	2811.49	2828.26	0.60	5271.41	2858.52	1.67	*1576.78*
Average				2428.17	2460.45	1.13	534.91	2449.49	0.69	*325.80*

Sadati et al. (2020). We utilized IBM ILOG CPLEX version 12.7.1 as the commercial MILP solver called from inside C#.

The KPIs obtained by CPLEX, TSH of Sadati et al. (2020) and VND + TSH are displayed in Table 4, where m_{eff} represents the number of operational depots. *CPLEX* (*UB*) and *CPLEX Gap* (%) indicate the best feasible solution (upper bound *UB*) found by CPLEX and the optimality gap of CPLEX, respectively. The relative gap between *CPLEX* (*UB*) and the objective value returned by the pertinent algorithm, which is calculated as $100 \times \frac{\text{Obj. Val.} - \textit{CPLEX}(\textit{UB})}{\textit{CPLEX}(\textit{UB})}$ is given in the columns with the header Gap (%). The proven optimal objective values and the respective zero gaps are indicated in **bold**. For those instances which cannot be solved to proven optimality by CPLEX, the best objective value and its respective Gap (%) are also indicated in **bold**. Moreover, the faster of the solution times for each instance among the three solution methods is indicated in *italic*.

Table 4 reveals that CPLEX could solve all 15 instances of pr01 to proven optimality within the allowed time limit. VND + TSH and the former TSH developed in Sadati et al. (2020) attained the optimal solution in five and three of these instances, respectively. The average deviation of VND + TSH from the optimal objective value is measured at 0.33%. This deviation amounts to 1.05% in TSH. When it comes to the solution speed, needless to say, both heuristics outpace CPLEX which spends 385.20 s in average until optimality is reached. Among the two, VND + TSH leads TSH with an average CPU time of 50.96 s against 54.24 s. CPLEX cannot report proven optimality in the remainder of our SMDVRP test bed which comprises 26 instances of pr03. It achieves the best solution in 10 of them. The same KPI for our proposed VND + TSH is 15, whereas that is just one for the TSH. VND + TSH produced an average deviation of 0.09% from *CPLEX* (*UB*) running for 103.38 s in average. On the other hand, the average deviation of TSH is as high as 2.46%, but its average CPU time requirement is relatively less, measuring 85.92 s.

Considering the entire SMDVRP test bed, we observe from Table 4 that VND + TSH beats the former TSH of Sadati et al. (2020) in solution accuracy at the expense of about 10 s in the average CPU time. In 21 out of 41 instances VND + TSH yields the same or a better solution than CPLEX. The former TSH can do so in only four instances. Being satisfied with the solution quality and efficiency of VND + TSH in MDVRP and SMDVRP test instances, we decided to employ it in our further experiments.

4.2. Results obtained on 3LRI-SMDVRP

In order to evaluate the performance of our comprehensive solution methodology SRE on the 3LRI-SMDVRP we built an initial test bed of 39 instances by taking subsets of customers found in three standard MDVRP instances, namely p01, p02 and pr01. The number of depots in these instances is equal to 4. The parameter u is fixed to 1, and the parameter r varies between 1 and 3. For each one of p01 and p02, we take 30, 25 and 20 customers into consideration with $u = 1$ and $r = 1, 2, 3$. This way we obtain $2 \times (3 \times 3) = 18$ distinct 3LRI-SMDVRP instances. For pr01, the numbers of customers considered are 48, 45, 40, 35, 30, 25 and 20 where each case produces 3 instances with $u = 1$ and $r = 1, 2, 3$. This gives rise to $(7 \times 3) = 21$ distinct 3LRI-SMDVRP instances generated from pr01. Hence, we experiment with a test bed of 39 instances in total. The unit outsourcing cost in each instance is assigned to the values shown in Table 6. Since the best solutions of these instances are unknown, we compare the

Table 4
Comparison of CPLEX and VND + TSH on 41 generated SMDVRP instances.

								TSH in Sadati et al. (2020)			The proposed VND + TSH		
Inst.	n	m	m_{eff}	Active depots	*CPLEX* (*UB*)	*CPLEX Gap* (%)	*CPLEX time* (*s*)	Obj. Val.	Gap (%)	Time (*s*)	Obj. Val.	Gap (%)	Time (s)
pr01	48	4	1	1	**753.59**	**0.00**	54.28	756.43	0.38	*38.90*	**753.59**	**0.00**	39.89
				2	**794.78**	**0.00**	330.34	800.60	0.73	79.19	799.98	0.65	*47.86*
				3	**768.89**	**0.00**	*27.23*	804.68	4.66	95.83	**768.89**	**0.00**	43.26
				4	**752.30**	**0.00**	*27.81*	778.56	3.49	46.99	756.92	0.61	64.34
	48	4	2	1, 2	**742.78**	**0.00**	102.78	744.37	0.21	*33.70*	**742.78**	**0.00**	68.45
				1, 3	**701.02**	**0.00**	60.89	**701.02**	**0.00**	75.28	**701.02**	**0.00**	*57.38*
				1, 4	**723.05**	**0.00**	729.58	728.92	0.81	*37.94*	726.50	0.48	45.88
				2, 3	**732.56**	**0.00**	1982.81	743.79	1.53	58.84	737.55	0.68	*45.54*
				2, 4	**725.94**	**0.00**	598.14	735.78	1.36	*58.49*	729.64	0.51	59.86
				3, 4	**719.18**	**0.00**	119.64	727.69	1.18	86.34	722.81	0.50	*41.76*
	48	4	3	1, 2, 3	**698.81**	**0.00**	203.02	**698.81**	**0.00**	*44.65*	701.02	0.32	68.28
				1, 2, 4	**717.22**	**0.00**	1260.97	**717.22**	**0.00**	*37.00*	**717.22**	**0.00**	52.95
				1, 3, 4	**693.06**	**0.00**	102.55	699.29	0.90	*41.03*	698.89	0.84	50.79
				2, 3, 4	**695.94**	**0.00**	85.56	697.64	0.24	46.42	697.32	0.20	*38.04*
	48	4	4	1, 2, 3, 4	**687.23**	**0.00**	92.39	688.60	0.20	*33.07*	688.60	0.20	40.07
p03	75	5	2	1, 2	737.24	7.91	10800.00	745.42	1.11	125.55	**734.94**	**−0.31**	*96.65*
				1, 3	**726.76**	**6.07**	10800.00	756.95	4.15	*73.17*	735.33	1.18	119.30
				1, 4	**737.56**	**5.44**	10800.00	762.99	3.45	116.98	752.36	2.01	*71.16*
				1, 5	**741.05**	**7.60**	10800.00	760.53	2.63	110.58	749.45	1.13	*103.27*
				2, 3	744.58	7.95	10800.00	769.13	3.30	*76.31*	**736.48**	**−1.09**	174.99
				2, 4	725.50	6.69	10800.00	746.61	2.91	90.16	**723.12**	**−0.33**	*72.51*
				2, 5	**789.39**	**7.83**	10800.00	812.35	2.91	*82.01*	793.89	0.57	139.45
				3, 4	739.38	6.89	10800.00	771.19	4.30	*87.15*	**733.92**	**−0.74**	166.04
				3, 5	**737.79**	**7.23**	10800.00	786.51	6.60	168.08	755.97	2.46	*95.60*
				4, 5	**793.06**	**6.91**	10800.00	827.69	4.37	*80.16*	819.19	3.29	82.04
	75	5	3	1, 2, 3	682.34	7.14	10800.00	701.59	2.82	*89.02*	**682.23**	**−0.02**	91.53
				1, 2, 4	701.00	9.03	10800.00	712.11	1.58	*54.08*	**698.20**	**−0.40**	100.82
				1, 2, 5	**694.80**	**7.83**	10800.00	717.85	3.32	123.12	694.94	0.02	*51.13*
				1, 3, 4	**691.89**	**6.80**	10800.00	711.70	2.86	*50.72*	708.76	2.44	73.48
				1, 3, 5	689.38	9.43	10800.00	706.23	2.44	*53.04*	**684.29**	**−0.74**	149.62
				1, 4, 5	**724.28**	**9.80**	10800.00	736.42	1.68	*80.76*	724.48	0.03	88.26
				2, 3, 4	658.43	6.95	10800.00	676.62	2.76	*52.37*	**655.82**	**−0.40**	135.56
				2, 3, 5	678.64	7.86	10800.00	705.37	3.94	*90.69*	**675.90**	**−0.40**	146.45
				2, 4, 5	703.70	7.90	10800.00	717.92	2.02	*55.36*	**699.58**	**−0.59**	113.28
				3, 4, 5	**686.65**	**7.74**	10800.00	710.17	3.43	*96.94*	695.15	1.24	116.21
	75	5	4	1, 2, 3, 4	650.09	7.19	10800.00	653.96	0.60	129.72	**648.45**	**−0.25**	*66.21*
				1, 2, 3, 5	650.78	8.25	10800.00	651.03	0.04	*88.87*	**645.73**	**−0.78**	106.33
				1, 2, 4, 5	680.49	9.51	10800.00	687.99	1.10	*57.80*	**670.68**	**−1.44**	83.67
				1, 3, 4, 5	670.47	9.17	10800.00	682.50	1.79	*56.94*	**667.18**	**−0.49**	59.90
				2, 3, 4, 5	654.19	9.39	10800.00	**635.63**	**−2.84**	*53.53*	**636.18**	**−2.75**	72.17
	75	5	5	1, 2, 3, 4, 5	626.58	6.77	10800.00	631.19	0.74	*90.85*	**617.92**	**−1.38**	112.22
Avg.					**712.74**	4.91	–	726.85	1.94	*74.33*	**714.22**	**0.18**	84.20

VND + TSH results with those obtained by CPLEX within a time limit of 3 h.

The comparison results are provided in Table 5 where the columns under CPLEX and VND + TSH show IDs of the protected depots, IDs of the interdicted depots, the objective value, CPU time in seconds, and the number of times CPLEX or VND + TSH is called, respectively. The last column indicates the relative gap between CPLEX and VND + TSH. In all CPLEX results, the term "*# Calls*" is equal to $\binom{m}{r}$.

CPLEX solves the LLP in all 3LRI-SMDVRP instances to optimality within the allowed time of three hours. This implies that using our proposed smart exhaustive enumeration (SEE) we can obtain the optimal protection and interdiction patterns along with the optimal objective values for all 39 instances. We observe from Table 5 that SRE with VND + TSH can find optimal protection and interdiction patterns in 36 instances. In 17 of these 36 instances it also yields the optimal objective value (**bolded** figures). SRE with VND + TSH attains an average percent deviation of 0.30% from the optimal objective value found by SEE using CPLEX. The overall solution accuracy and speed of VND + TSH incorporated into the two reduction methods Reduction 1 and Reduction 2 is promising on all 39 3LRI-SMDVRP instances. Its average CPU time is 70.88 s and the average number of calls is 3.38, whilst CPLEX requires 527.84 s to converge to the same respective solution with 4.67 calls in average.

Our second test bed contains 287 3LRI-SMDVRP instances which are constructed from 33 MDVRP benchmark instances existing in the literature. These 33 instances differ in the number of depots and customers. The parameters of protection and interdiction resources, namely u and r vary between 1 and 5 with increments of one. The value of the outsourcing cost per unit demand, namely c^o is given in Table 6 for each instance. We tried to assign c^o values in such a way that neither all customers are outsourced (which would be caused by a too low c^o) nor no customer is outsourced at all (which would be caused by a too high c^o) in the resulting solution.

The results of the two proposed reduction techniques applied to 33 3LRI-SMDVRP instances all with $r = 1$ and $u = 1$ are given in Table 7. We report the solution value (*ObjVal*) and solution time for each instance. The column R_i shows which reduction technique is used (R_1: Reduction 1, R_2: Reduction 2). The ID columns indicate the IDs of the protected depots (*u_ID*)and interdicted depots(*r_ID*).

The column header *"MNPP"* indicates the maximum number of possible patterns $\binom{m}{r}$. The column header *"VND + TSH Calls"* stands for the number of times the VND + TSH algorithm is called by each reduction technique. Detailed results for different levels of

Table 5
SEE with CPLEX versus SRE with VND + TSH on 39 3LRI-SMDVRP instances.

Inst.	m	n	u	r	SEE with CPLEX					SRE with VND + TSH					Gap (%)
					u_ID	r_ID	Obj. Val.*	Time (s)	# Calls	u_ID	r_ID	Obj. Val.	Time (s)	# Calls	
p01	4	30	1	1	**4**	**2**	375.12	641.23	4	**4**	**2**	376.76	94.23	3	0.44
				2	**2**	**3, 4**	396.17	412.93	6	**2**	**3, 4**	398.07	101.26	5	0.48
				3	**2**	**1, 2, 3**	**407.87**	68.25	4	**2**	**1, 2, 3**	**407.87**	25.61	1	**0.00**
		25	1	1	4	2	321.19	52.14	4	2	4	331.80	39.59	3	3.30
				2	**1**	**3, 4**	344.06	84.25	6	**1**	**3, 4**	347.18	63.23	5	0.91
				3	**2**	**1, 3, 4**	357.70	18.52	4	**2**	**1, 3, 4**	358.66	6.94	2	0.27
		20	1	1	**4**	**3**	263.43	28.52	4	**4**	**3**	264.52	13.36	4	0.41
				2	1	3, 4	291.03	35.26	6	3	1, 2	292.02	24.69	4	0.34
				3	**2**	**1, 3, 4**	291.02	15.62	4	**2**	**1, 3, 4**	292.48	10.85	2	0.50
p02	4	30	1	1	**1**	**4**	**308.34**	284.90	4	**1**	**4**	**308.34**	84.47	4	**0.00**
				2	**2**	**1, 4**	**316.45**	1577.31	6	**2**	**1, 4**	**316.45**	185.59	6	**0.00**
				3	**2**	**1, 3, 4**	**327.86**	2066.99	4	**2**	**1, 3, 4**	**327.86**	28.41	1	**0.00**
		25	1	1	**1**	**4**	**273.69**	122.23	4	**1**	**4**	**273.69**	52.13	4	**0.00**
				2	**2**	**1, 3**	**283.97**	486.16	6	**2**	**1, 3**	**283.97**	82.23	6	**0.00**
				3	**2**	**1, 3, 4**	**294.11**	641.32	4	**2**	**1, 3, 4**	**294.11**	11.12	1	**0.00**
		20	1	1	**4**	**1**	**240.85**	35.15	4	**4**	**1**	**240.85**	26.35	4	**0.00**
				2	**1**	**3, 4**	**249.41**	51.26	6	**1**	**3, 4**	**249.41**	38.14	6	**0.00**
				3	**2**	**1, 3, 4**	**249.41**	10.26	4	**2**	**1, 3, 4**	**249.41**	5.45	1	**0.00**
pr01	4	48	1	1	**3**	**4**	698.81	1652.10	4	**3**	**4**	701.02	145.43	4	0.32
				2	**4**	**1, 3**	725.94	3593.84	6	**4**	**1, 3**	729.64	193.47	5	0.51
				3	4	1, 2, 3	752.30	439.66	4	1	2, 3, 4	753.59	72.38	2	0.17
		45	1	1	**3**	**4**	648.19	2005.94	4	**3**	**4**	650.47	193.17	4	0.35
				2	**3**	**1, 4**	671.16	2043.78	6	**3**	**1, 4**	672.59	268.23	5	0.21
				3	**4**	**1, 2, 3**	701.88	233.44	4	**4**	**1, 2, 3**	703.29	179.49	3	0.20
		40	1	1	**3**	**2**	580.01	543.36	4	**3**	**2**	582.42	154.27	4	0.42
				2	**3**	**1, 2**	603.57	1183.67	6	**3**	**1, 2**	605.49	195.03	5	0.32
				3	**4**	**1, 2, 3**	639.35	1501.67	4	**4**	**1, 2, 3**	642.59	102.62	3	0.51
		35	1	1	**3**	**4**	537.07	161.94	4	**3**	**4**	539.12	58.25	2	0.38
				2	**3**	**2, 4**	541.34	201.41	6	**3**	**2, 4**	544.05	142.29	5	0.50
				3	**4**	**1, 2, 3**	**571.40**	128.85	4	**4**	**1, 2, 3**	**571.40**	60.07	2	**0.00**
		30	1	1	**3**	**4**	481.48	54.36	4	**3**	**4**	482.42	33.79	2	0.20
				2	**3**	**2, 4**	492.43	78.93	6	**3**	**2, 4**	495.55	41.26	4	0.63
				3	**3**	**1, 2, 4**	506.55	20.06	4	**3**	**2, 4**	508.55	4.57	2	0.39
		25	1	1	**3**	**4**	**422.48**	18.42	4	**3**	**4**	**422.48**	17.67	2	**0.00**
				2	**3**	**1, 2**	**435.72**	35.26	6	**3**	**1, 2**	**435.72**	2.86	3	**0.00**
				3	**3**	**1, 2, 4**	**447.57**	18.59	4	**3**	**1, 2, 4**	**447.57**	2.41	4	**0.00**
		20	1	1	**3**	**2**	**290.99**	6.54	4	**3**	**2**	**290.99**	1.58	2	**0.00**
				2	**3**	**1, 2**	**306.58**	25.26	6	**3**	**1, 2**	**306.58**	1.54	3	**0.00**
				3	**3**	**1, 2, 4**	**308.07**	6.24	4	**3**	**1, 2, 4**	**308.07**	0.32	4	**0.00**
Avg.							434.73	527.84	4.67			436.08	70.88	3.38	0.30

* Optimal values

Table 6
The unit outsourcing cost and the share of total outsourcing cost in the final solution when $u = r = 0$.

Instance	c^o	Ratio (%)	Problem name	c^o	Ratio (%)	Problem name	c^o	Ratio (%)
p01	1	10.00	p12	12	10.00	p23	15	10.00
p02	1	14.00	p13	10	17.50	pr01	1.5	20.83
p03	0.8	10.67	p14	10	30.00	pr02	1.5	22.92
p04	1	17.00	p15	14	10.00	pr03	1.5	20.14
p05	1	9.00	p16	15	4.38	pr04	1	19.79
p06	1	12.00	p17	15	10.00	pr05	2	5.42
p07	1	10.00	p18	6	30.00	pr06	2	7.99
p08	0.7	10.04	p19	15	2.50	pr07	2	18.06
p09	0.5	8.43	p20	15	10.00	pr08	1.5	9.03
p10	0.5	12.05	p21	14	10.00	pr09	1	26.85
p11	0.5	10.04	p22	15	2.22	pr10	2	4.51

$u = (1, \cdots, 5)$ and $r = (1, \cdots, 5)$ are available for download from the URL address http://myweb.sabanciuniv.edu/msadati/3LRI-SMDVRP.

The performance of the proposed solution methods is summarized in the following four tables in terms of the average CPU time and average counts of VND + TSH calls for different u and r values. Tables 8 and 9 show the average CPU time for each method. We observe that for a given value of$u$ (r) the average CPU time increases as the cardinality of interdiction (protection) resources, namely r (u) increases.

Similar tables are presented for the average number of VND + TSH calls. In Tables 10 and 11 the average MNPP and average VND + TSH call counts are reported in the first and second columns, respectively, for a given pair of u and r. The obtained results prove the effectiveness of the methods for solving the 3LRI-SMDVRP with a lesser number of VND + TSH calls instead of solving

Table 7
Results of 3LRI-SMDVRP instances with $u = 1,\ r = 1$.

Instance	n	m	R_i	u_ID	r_ID	Obj_Value	$MNPP$	$VND + TSH\ Calls$	$Time\ (s)$
p01	50	4	R_1	2	4	578.82	4	4	285.28
p02	50	4	R_1	2	1	461.69	4	4	308.65
p03	75	5	R_1	3	2	667.18	5	4	365.07
p04	100	2	R_2	1	2	1146.34	2	1	108.93
p05	100	2	R_2	1	2	852.06	2	1	144.36
p06	100	3	R_1	1	3	975.89	3	3	578.40
p07	100	4	R_1	4	1	914.08	4	4	829.94
p08	249	2	R_2	1	2	5521.92	2	2	725.61
p09	249	3	R_1	1	2	4654.54	3	2	641.53
p10	249	4	R_1	2	1	4078.68	4	2	558.33
p11	249	5	R_1	2	4	3865.18	5	5	1627.87
p12	80	2	R_2	2	1	1771.21	2	2	171.77
p13	80	2	R_2	2	1	3813.42	2	2	135.37
p14	80	2	R_2	2	1	4074.55	2	2	130.65
p15	160	4	R_1	2	3	2959.85	4	4	438.18
p16	160	4	R_1	3	2	4734.88	4	4	336.99
p17	160	4	R_1	2	1	5174.04	4	2	168.58
p18	240	6	R_1	5	6	3850.64	6	2	318.69
p19	240	6	R_1	2	6	5924.58	6	4	846.72
p20	240	6	R_1	6	2	6399.78	6	2	375.23
p21	360	9	R_1	3	7	6144.41	9	5	5302.09
p22	360	9	R_1	5	7	7847.96	9	4	2279.97
p23	360	9	R_1	1	5	8357.61	9	4	2427.22
pr01	48	4	R_1	3	4	701.02	4	4	145.43
pr02	96	4	R_1	1	4	985.81	4	3	450.17
pr03	144	4	R_1	2	3	1638.71	4	3	302.68
pr04	192	4	R_1	4	3	1746.94	4	4	651.83
pr05	240	4	R_1	1	4	2493.55	4	3	1921.06
pr06	288	4	R_1	3	2	2720.58	4	3	2090.29
pr07	72	6	R_1	4	2	1013.35	6	3	253.52
pr08	144	6	R_1	5	1	1612.14	6	4	440.18
pr09	216	6	R_1	3	4	1864.80	6	6	1055.36
pr10	288	6	R_1	4	1	2867.17	6	5	3611.38
Average			R_1				5.1	3.6	1059.65
			R_2				2.0	1.7	236.11
			Overall				4.5	3.2	909.92

Table 8
Average CPU time in seconds when $u + r < m$.

		r				
		1	2	3	4	5
u	1	1059.65	1255.09	1983.25	3314.69	9308.56
	2	1272.24	2840.96	4418.66	17843.54	33234.18
	3	2197.20	4101.85	18797.71	45104.76	87514.91
	4	2605.89	10735.36	37339.84	87712.03	–[a]
	5	6106.20	19744.42	62584.85	–	–

[a] No such instance exists in our test bed for the given values of $u$and r.

Table 9
Average CPU time in seconds when $u + r = m$.

		r				
		1	2	3	4	5
u	1	236.11	375.72	278.87	272.05	571.96
	2	298.69	342.79	199.66	1899.88	–
	3	399.95	883.79	1722.35	–	–
	4	160.56	1324.71	–	–	7918.20
	5	722.28	–	–	5593.23	–

all possible protection and interdiction patterns exhaustively. In all results, the average number of VND + TSH calls is smaller than the average MNPP. The largest MNPP is found in three instances that have been derived from p21, p22 and p23 where we have $m = 9$ and$r$ =5. Therefore, MNPP is given by $\binom{9}{5}$ =126. Tables 10 and 11 reveal that the final solutions for these instances can be obtained with much fewer VND + TSH calls than 126.

4.3. Parametric tuning of the parameter ψ

Recall that Reduction 2 helps to solve only those combinations CWH + iMCA solutions of which can be improved using VND + TSH by at least $\psi \times 100\%$. Choosing an appropriate value for the tolerance parameter ψ is critical because too low a value decreases the number of VND + TSH calls, but it increases the risk of obtaining an inferior solution at the end of Reduction 2. Too high a value, on the

Table 10
Average MNPP and VND + TSH calls when $u + r < m$.

Average MNPP, Average # of VND + TSH calls											
		r									
		1		2		3		4		5	
u	1	5.1	3.6	12.4	5.3	34.3	6.2	48.3	6.3	126	7.3
	2	5.2	4.1	19.4	8.6	39.2	12.4	126	15.7	126	26.3
	3	6.6	5.3	21.3	10.9	84	20.7	126	41.7	126	69.0
	4	6.9	6.1	36	13.3	84	40.7	126	81.7	–	–
	5	9	8.3	36	25.0	84	68.7	–	–	–	–

Table 11
Average MNPP and VND + TSH calls when $u + r = m$.

Average MNPP, Average # of VND + TSH calls											
		r									
		1		2		3		4		5	
u	1	2	1.7	3	2	4	1.8	5	1.5	6	3.3
	2	3	1.0	6	1.9	10	1.0	15	6.1	–	–
	3	4	2.1	10	3.0	20	6.0	–	–	–	–
	4	5	1.0	15	5.0	–	–	–	–	126	7.3
	5	6	3.6	–	–	–	–	126	5.3	–	–

other hand, guarantees a superior solution with a lower LLP objective value for the defender. However, it increases the number of VND + TSH calls which may be equal to $\binom{m}{r}$ in the worst case. We try to set the value of ψ in such a way that a superior solution same as that of Reduction 1 is obtained with a lesser number of VND + TSH calls.

In order to determine a suitable value for the tolerance parameter ψ, we select three 3LRI-SMDVRP test instances that have been derived from the MDVRP benchmark instances p07 (m = 4), p03 (m =5) and pr08 (m = 6). We solved all possible cases of $u + r = m$ with both Reduction 1 and Reduction 2 using different values of ψ = 0.02, 0.04, 0.06 and 0.08. The results are presented in Tables 12–14. For each reduction technique we report the best objective value of the trilevel problem, the number of VND + TSH calls, and CPU time in seconds in the first, second and third columns, respectively. Column "Reduction 1" in the tables shows the best solution which is obtained after solving all possible $\binom{m}{r}$ patterns.

The **bold** values in Tables 12–14 show the best solutions obtained by each reduction technique. The results indicate that when the tolerance parameter ψ is set to a small value, namely 0.02, the number of VND + TSH calls reduces compared to that number in Reduction 1. However, inferior solutions are attained (e.g., in pr08 with $u = 2$, $r = 4$). The three values of ψ = 0.04, 0.06 and 0.08 result in the same solutions as Reduction 1. The value ψ = 0.04 produces the best solutions with fewer VND + TSH calls (thus, in less CPU time) compared to ψ = 0.06 and 0.08. Therefore, in our computational study we adopted ψ = 0.04.

5. Conclusion and future work

In this paper we introduce a trilevel optimization problem for the protection of the most critical depots in a multi-depot vehicle routing network. The problem is modelled as a defender-attacker-defender game from the perspective of the defender who needs to protect a limited number of depots on an existing network against interdiction by an adversary agent whom we designate as the attacker. The attacker's objective is to inflict the maximum disruption on this network by annihilating a certain number of unprotected depots beyond repair. We refer to this problem as the trilevel r-interdiction selective multi-depot vehicle routing problem (3LRI-SMDVRP). The defender is the decision maker in the upper level problem (ULP) who decides which depots to protect. In the middle level problem (MLP), the attacker chooses r depots to interdict among the unprotected ones. Finally, in the lower level problem (LLP), the decision maker is again the defender who optimizes the vehicle routes and thereby selects which customers are to be visited and which ones to be outsourced to a third-party logistics company. As a solution approach to this trilevel discrete optimization problem, we propose two reduction techniques (Reduction 1 and Reduction 2) which solve the ULP and MLP through smart exhaustive enumeration. For the LLP we implement a hybrid metaheuristic method combining Variable Neighborhood Descent and Tabu Search Heuristic (VND + TSH) techniques adapted to the selective multi-depot VRP (SMDVRP). Our preliminary numerical experiments validate the performance of VND + TSH on 33 standard MDVRP and 41 randomly generated SMDVRP instances. We then carry out experiments on two separate test beds of 39 and 287 3LRI-SMDVRP instances that were con-

Table 12
Performance of Reduction 1 and 2 on p07 (m = 4).

VND + TSH solution, # of VND + TSH calls, CPU time (s)																
					Reduction 2 ψ											
u	r	Reduction 1			0.02			0.04			0.06			0.08		
1	3	**1146.34**	4	684.59	**1146.34**	2	381.91	**1146.34**	2	382.55	**1146.34**	3	551.24	**1146.34**	4	662.24
2	2	**967.69**	6	1329.73	**967.69**	1	151.58	**967.69**	1	152.86	**967.69**	3	818.64	**967.69**	4	980.54
3	1	**901.15**	4	1164.20	901.65	1	237.15	**901.15**	2	492.01	**901.15**	2	493.08	**901.15**	3	845.11

Table 13
Performance of Reduction 1 and 2 on p03 (m = 5).

VND + TSH solution, # of VND + TSH calls, CPU time (s)																
					Reduction 2 ψ											
u	r	Reduction 1			0.02			0.04			0.06			0.08		
1	4	**798.93**	5	495.25	**798.93**	1	181.02	**798.93**	1	181.21	**798.93**	1	181.15	**798.93**	1	181.71
2	3	**731.41**	10	1478.89	**731.41**	1	123.34	**731.41**	1	123.67	**731.41**	1	123.51	**731.41**	5	814.04
3	2	**663.73**	10	1246.95	**663.73**	1	96.54	**663.73**	1	96.46	**663.73**	1	96.41	**663.73**	3	339.31
4	1	**636.18**	5	601.75	**636.18**	1	89.03	**636.18**	1	89.17	**636.18**	1	88.96	**636.18**	3	410.91

Table 14
Performance of Reduction 1 and 2 on p08 (m = 6).

VND + TSH solution, #VND + TSH calls, CPU time (s)																
					Reduction 2 ψ											
u	r	Reduction 1			0.02			0.04			0.06			0.08		
1	5	**1908.85**	6	585.84	**1908.85**	1	88.99	**1908.85**	1	88.84	**1908.85**	1	88.64	**1908.85**	1	88.61
2	4	**1702.82**	15	1918.03	1749.22	1	183.41	**1702.82**	3	464.76	**1702.82**	3	464.61	**1702.82**	5	748.68
3	3	**1613.69**	20	2620.21	1614.48	1	212.25	**1613.69**	2	344.92	**1613.69**	2	345.63	**1613.69**	8	1164.51
4	2	**1574.47**	15	2117.77	1586.18	1	121.91	**1574.47**	4	489.76	**1574.47**	7	1082.38	**1574.47**	11	1574.39
5	1	**1544.22**	6	952.66	1566.05	1	132.81	**1544.22**	2	332.68	**1544.22**	2	334.43	**1544.22**	5	833.22

structed from 33 standard MDVRP instances with different parameters for the outsourcing cost per unit demand, the number of depots to be protected by the defender, and the number of depots to be interdicted by the attacker.

The computational results attest to the effectiveness of the proposed reduction techniques for solving the 3LRI-SMDVRP. We accomplish to solve the trilevel problem with a smaller number of VND + TSH calls compared to the approach of exploring and solving all possible patterns. In our future work, we are planning to incorporate depot location decisions into the ULP of the 3LRI-SMDVRP, and treat it as location-protection-interdiction-routing problem. In this extended problem, the defender has to determine where to open depots and which of them to protect simultaneously. As a solution approach, a matheuristic method could be developed that would combine exact and heuristic techniques to obtain near-optimal solutions within acceptable times. Differentiation among depots with respect to their protection costs under a hard budget constraint could also be a realistic extension worthy of consideration.

CRediT authorship contribution statement

Mir Ehsan Hesam Sadati: Conceptualization, Methodology, Software, Writing - original draft, Visualization, Investigation, Data curation, Validation, Formal analysis. **Deniz Aksen:** Conceptualization, Methodology, Supervision, Resources, Writing - review & editing, Validation, Formal analysis. **Necati Aras:** Conceptualization, Methodology, Writing - review & editing, Validation.

Acknowledgment

The authors would like to thank two anonymous reviewers for their constructive and instrumental feedback on the original draft of this paper. Their suggestions and comments helped us to amend the paper and improve its clarity by a great deal.

Appendix A

Examples illustrating Reduction 1 and Reduction 2

As an example for Reduction 1 (CASE I), consider the instance p03 which has 75 customers and 5 depots. The value of the outsourcing cost (c^o) is equal to 0.8, and both the number of protected depots (u) and the number of interdicted depots (r) are set to one. There are $\binom{5}{1} = 5$ protection patterns from the defender's perspective and for each protection pattern there exist $\binom{4}{1} = 4$ interdiction patterns from attacker's viewpoint. In each interdiction pattern, attacker will select the patterns that give rise to biggest disruption in the network (highest objective value) and the defender's best decision in response is to select the best protection pattern which results in the least possible disruption in the network (minimizing the maximum disruption).

Table A1 shows all interdiction patterns for each protection pattern. For example, if the defender protects the first depot, the attacker can interdict one of the remaining depots (second, third, fourth or fifth depot). In this table the "Order" column indicates the order of calling VND + TSH for each interdiction pattern. Reduction 1 starts form solving all possible interdiction patterns of the attacker using CWH + iMCA when the first depot is protected. According to the solution of CWH + iMCA interdicting 3rd depot has the highest objective value (701.44) and this interdiction pattern is solved one more time with VND + TSH (improved to 686.70). Now this improved solution is compared against the second highest CWH + iMCA solution which has an objective value of 695.42 and suggested the interdiction of the 5th depot. Because it is less, all interdiction patterns whose CWH + iMCA objective value is greater than 686.70 will be solved once again with VND + TSH for the purpose of finding a better interdiction pattern. Note that each ULP and MLP solution pair produced during the iterations of Reduction 1 along with the associated LLP objective value coming from VND + TSH will be stored in a hash table for future access. After solving the relevant interdiction patterns with VND + TSH,

Table A1
CASE I example: P03 ($m = 5, u = r = 1$).

Problem Name	*u_ID*	*r_ID*	CWH + iMCA	Order	Calling VND + TSH?	VND + TSH	Best Interdiction Pattern
p03	1	2	686.86	3	Yes	667.18	(1, 3, 686.70)
		3	*701.44*	1	Yes	***686.70***	
		4	678.97	–	No[1]	–	
		5	695.42	2	Yes	646.06	
	2	1	648.16	–	No	–	(2, 3, 686.70)
		3	*701.44*	1	No (H)[2]	***686.70***	
		4	678.97	–	No	–	
		5	695.42	2	No (H)	646.06	
	3	1	648.16	–	No	–	**(3, 2, 667.18)**
		2	686.86	2	No (H)	***667.18***	
		4	678.97	3	Yes	645.73	
		5	*695.42*	1	No (H)	646.06	
	4	1	648.16	–	No	–	(4, 3, 686.70)
		2	686.86	3	No (H)	667.18	
		3	*701.44*	1	No (H)	***686.70***	
		5	695.42	2	No (H)	646.06	
	5	1	648.16	–	No	–	(5, 3, 686.70)
		2	686.86	2	No (H)	667.18	
		3	*701.44*	1	No (H)	***686.70***	
		4	678.97	–	No	–	

[1] No: No need to consider this pattern (its CWH + iMCA objective value is already less than VND + TSH solution.).
[2] No (H): No need to Call VND + TSH, the solution of this pattern is stored in Hash Table already.

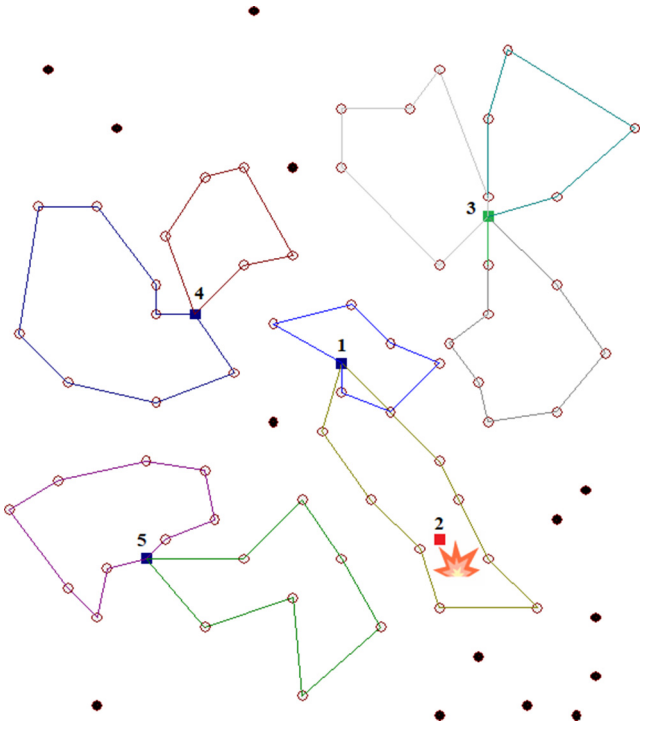


Fig. A1. Visual representation of final solutions for p03 with $u = r = 1$.

the best interdiction pattern will consist of the 3rd depot with an objective value of 686.70.

This process is continued for each protection pattern. The best interdiction patterns are presented in the last column of Table A1. The final solution consists of protecting the 3rd depot and interdicting the 2nd depot with a minimum objective value of 667.18. VND + TSH is called throughout the entire Reduction 1 procedure 4 times in total, which is less than $\binom{m}{r} = \binom{5}{1} = 5$. The best CWH + iMCA and VND + TSH solution for each interdiction pattern in Table A1 has been shown in *italic* and ***bolded italic***, respectively. Fig. A1 presents a visualization of the final solution where customers are shown by circles and depots by squares. The protected and interdicted depots and the outsourced customers are designated in green, red and black color, respectively.

As an example for Reduction 2 (CASE II), consider again the test instance p03 with $u = 3$, $r = 2$ $(u + r = m)$ and with the tolerance parameter $\psi = 0.04$. Table A2 shows all $\binom{5}{2}$ =10 protection and interdiction patterns. Reduction 2 starts by solving all patterns using CWH + iMCA, and selects the one with the minimum objective value of 691.21—protecting depots 2, 3 and 4 while interdicting depots 1 and 5. This pattern is solved one more time with VND + TSH for further improvement (improved to 663.73). Now this improved solution is compared against that of the remaining patterns obtained by CWH + iMCA. Because it is less than $\frac{\text{Obj.Val.(CWH+iMCA)}}{1.0+4.0\%}$, the algorithm terminates. It can be seen that no

Table A2
CASE II example: P03 ($m = 5, u = 3$, $r = 2$ and $\psi = 0.04$).

Problem Name	*u_ID*	*r_ID*	CWH + iMCA	$\frac{\text{CWH + iMCA}}{1+\psi}$	Order	VND + TSH
p03	1,2,3	4,5	730.13	702.05	–	–
	1,2,4	3,5	729.24	701.19	–	–
	1,2,5	3.4	735.22	706.94	–	–
	1,3,4	2,5	729.92	701.85	–	–
	1,3,5	2,4	707.19	679.99	–	–
	1,4,5	2,3	756.96	727.85	–	–
	2,3,4	**1,5**	*691.21*	664.63	1	**663.73**
	2,3,5	1,4	706.87	679.68	–	–
	2,4,5	1,3	734.41	706.16	–	–
	3,4,5	1,2	749.84	721.00	–	–

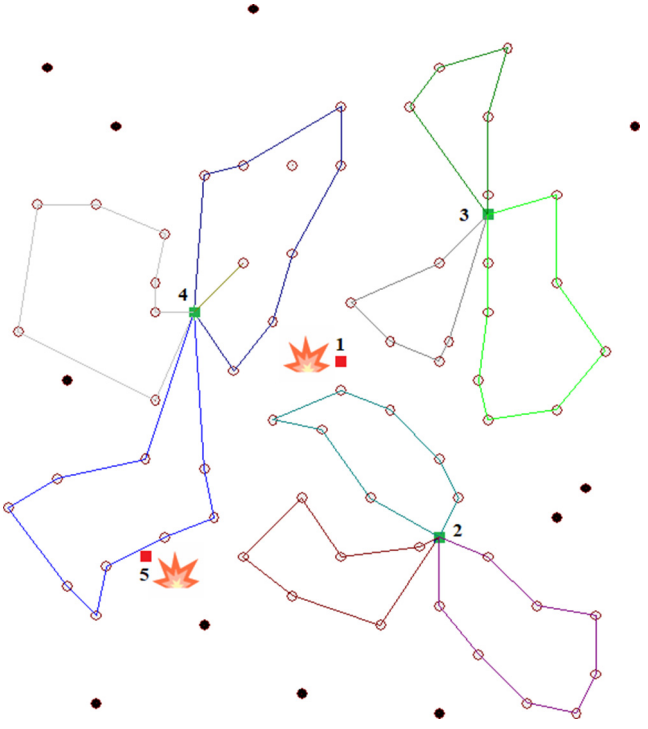


Fig. A2. Visual representation of final solutions for p03 with $u = 3$ and $r = 2$.

other pattern can yield a better (lower) objective value than 663.73 using VND + TSH. Therefore, the final solution consist of protecting depots 2, 3, and 4 and interdicting depots 1 and 5 with an objective value of 663.73. In order to find this solution VND + TSH is called only ***one*** time which is far less than $\binom{m}{r} = \binom{5}{2} = 10$. Fig. A2 presents a visualization of the final solution for this example. Note that the validation of Reduction 2 for this instance has been provided in Table 13.

Appendix B

The tabu search heuristic TSH

We execute TSH as a local search procedure embedded in our hybrid algorithm VND + TSH which solves the LLP of the 3LRI-SMDVRP heuristically. It is called in Step 13 of the VND + TSH (see Algorithm 3). Given the current solution $\bar{S}$ and the subset of admissible arcs *AdmissArc* associated with the current move type k of the main VND loop, TSH returns a local optimal solution. Algorithm 4 presents the pseudo code of TSH.

Algorithm 4. $TSH\left(\bar{S},\ k,\ t,\ \theta_{i_{N_k}},\ \rho_{i_{N_k}},\ \omega,\ AdmissArc\right)$

Additional Notation

$N_{sub}(S)$:	Subset of $N_k(S)$.
σ:	Number of solutions in $N_{sub}(S)$.
n_S:	Number of customers in solution S.
v_S:	Number of routes in solution S.

// Solution $\bar{S}$ and parameters $k, \theta_{i_{N_k}}$ and $\rho_{i_{N_k}}$ are inherited from iteration t of the parent algorithm VND + TSH.

1: Set $S' = \bar{S}$ and $Z(S') = Z\left(\bar{S}\right)$;
2: Generate $N_{sub}\left(\bar{S}\right)$ from $N_k\left(\bar{S}\right)$ based on *AdmissArc* and record $\sigma = \left|N_{sub}\left(\bar{S}\right)\right|$;
3: **For** each solution $\hat{S} \in N_{sub}\left(\bar{S}\right)$
4: Compute $Z'\left(\hat{S}\right)$ according to Eq. (31);
5: **If** $Z'\left(\hat{S}\right) \geqslant Z'\left(\bar{S}\right)$
6: Update $Z'\left(\hat{S}\right) = Z'\left(\hat{S}\right) + \sqrt{n_{\bar{S}} v_{\bar{S}}}\left(\frac{\rho_{i_{N_k}}}{t}\right) \times \left(Z'\left(\hat{S}\right) - Z'\left(\bar{S}\right)\right)$; // *Diversification*
7: **End If**
8: Record $Z'\left(\hat{S}\right)$ in an array of objective values denoted by $Z'\left(\hat{S}\right)[]$;
9: **End For**
10: Sort the array $Z'\left(\hat{S}\right)[]$ in ascending order;
// $Z'\left(\hat{S}\right)[1]$ *denotes the smallest objective value.*
11: Set index = 1 and Flag = TRUE;
12: **While** (Flag = TRUE **AND** index < σ)
13: Select the solution $\hat{S}$ stored in $Z'\left(\hat{S}\right)[index]$;
14: **If** $(t > \theta_{i_{N_k}})$ **OR** $(Z'\left(\hat{S}\right) < \omega)$ // *Check for the aspiration criterion*
15: Update $S' = \hat{S}$ and $Z(S') = Z\left(\hat{S}\right)$;
16: Update $\theta_{i_{N_k}} = t + \lambda$ and $\rho_{i_{N_k}} = \rho_{i_{N_k}} + 1$;
17: Set Flag = FALSE;
18: **Else**
19: Set index = index + 1;
20: **End If**
21: **End While**
22: Return S'

References

Akbari-Jafarabadi, M., Tavakkoli-Moghaddam, R., Mahmoodjanloo, M., Rahimi, Y., 2017. A tri-level *r*-interdiction median model for a facility location problem under imminent attack. Comput. Ind. Eng. 114, 151–165.

Aliakbarian, N., Dehghanian, F., Salari, M., 2015. A bi-level programming model for protection of hierarchical facilities under imminent attacks. Comput. Oper. Res. 64, 210–224.

Aksen, D., Akca, S.Ş., Aras, N., 2014. A bilevel partial interdiction problem with capacitated facilities and demand outsourcing. Comput. Oper. Res. 41, 346–358.

Aksen, D., Aras, N., 2012. A bilevel fixed charge location model for facilities under imminent attack. Comput. Oper. Res. 39 (7), 1364–1381.

Aksen, D., Aras, N., Piyade, N., 2013. A bilevel *p*-median model for the planning and protection of critical facilities. J. Heuristics 19 (2), 373–398.

Aksen, D., Piyade, N., Aras, N., 2010. The budget constrained *r*-interdiction median problem with capacity expansion. Cent. Eur. J. Oper Res. 18 (3), 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., Tekin, M.T., 2011. Selective multi-depot vehicle routing problem with pricing. Trans. Res. Part C 19 (5), 866–884.

Arroyo, J.M., Galiana, F.D., 2005. On the solution of the bilevel programming formulation of the terrorist threat problem. IEEE Trans. Power Syst. 20 (2), 789–797.

Brown, G. G., Carlyle, W. M., Salmeron, J., and Wood, K., 2005. Analyzing the vulnerability of critical infrastructure to attack and planning defenses. In Emerging Theory, Methods, and Applications (pp. 102-123). INFORMS.

Bidgoli, M.M., Kheirkhah, A., 2018. An arc interdiction vehicle routing problem with information asymmetry. Comput. Ind. Eng. 115, 520–531.

Church, R.L., Scaparra, M.P., Middleton, R.S., 2004. Identifying critical infrastructure: the median and covering facility interdiction problems. Ann. Assoc. Am. Geogr. 94 (3), 491–502.

Church, R.L., Scaparra, M.P., 2007. Protecting critical assets: the *r*-interdiction median problem with fortification. Geogr Anal 39 (2), 129–146.

Clarke, G., Wright, J.W., 1964. Scheduling of vehicles from a central depot to a number of delivery points. Oper. Res. 12 (4), 568–581.

Cordeau, J.F., Gendreau, M., Laporte, G., 1997. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks 30 (2), 105–119.

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., Iori, M., Laporte, G., Martello, S., 2008. A Tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. Networks 51 (1), 4–18.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. Comput. Oper. Res. 13 (5), 533–549.
Gouveia, L., 1995. A result on projection for the vehicle routing problem. Eur. J. Oper. Res. 85 (3), 610–624.
Israeli, E., Wood, R.K., 2002. Shortest-path network interdiction. Networks 40 (2), 97–111.
Kheirkhah, A., Navidi, H., Bidgoli, M.M., 2016a. A bi-level network interdiction model for solving the hazmat routing problem. Int. J. Prod. Res. 54 (2), 459–471.
Kheirkhah, A., Navidi, H., Bidgoli, M.M., 2016b. An improved benders decomposition algorithm for an arc interdiction vehicle routing problem. IEEE Trans. Eng. Manage. 63 (2), 259–273.
Lenstra, J.K., Rinnooy Kan, A.R., 1981. Complexity of vehicle routing and scheduling problems. Networks 11 (2), 221–227.
Lozano, L., Smith, J.C., Kurz, M.E., 2017. Solving the traveling salesman problem with interdiction and fortification. Oper. Res. Lett. 45 (3), 210–216.
Mahmoodjanloo, M., Parvasi, S.P., Ramezanian, R., 2016. A tri-level covering fortification model for facility protection against disturbance in r-interdiction median problem. Comput. Ind. Eng. 102, 219–232.
McMasters, A.W., Mustin, T.M., 1970. Optimal interdiction of a supply network. Nav. Res. Logist. Q 17 (3), 261–268.
Mladenović, N., Hansen, P., 1997. Variable neighborhood search. Comput. Oper. Res. 24 (11), 1097–1100.
Moore, J.T., Bard, J.F., 1990. The mixed integer linear bilevel programming problem. Oper. Res. 38 (5), 911–921.
Salmeron, J., Wood, K., Baldick, R., 2004. Analysis of electric grid security under terrorist threat. IEEE Trans. Power Syst. 19 (2), 905–912.
San Martin, P.A., 2007. Tri-level optimization models to defend critical infrastructure. Master's thesis, Naval Postgraduate School, Monterey, California. https://apps.dtic.mil/docs/citations/ADA473701 (accessed 28 March 2020).
Sadati, M.E.H., Aksen, D., Aras, N., 2020. The *r*-interdiction selective multi-depot vehicle routing problem. Int. Trans. Oper Res. 27 (2), 835–866.
Scaparra, M.P., Church, R.L., 2008. A bilevel mixed-integer program for critical infrastructure protection planning. Comput. Oper. Res. 35 (6), 1905–1923.
Scaparra, M.P., Church, R., 2012. Protecting supply systems to mitigate potential disaster: a model to fortify capacitated facilities. Int. Regional Sci. Rev. 35 (2), 188–210.
Stenger, A., Vigo, D., Enz, S., Schwind, M., 2013a. An adaptive variable neighborhood search algorithm for a vehicle routing problem arising in small package shipping. Transp. Sci. 47 (1), 64–80.
Stenger, A., Schneider, M., Goeke, D., 2013b. The prize-collecting vehicle routing problem with single and multiple depots and non-linear cost. EURO J. Transp. Logist. 2 (1–2), 57–87.
Toth, P., Vigo, D., 2003. The granular tabu search and its application to the vehicle-routing problem. INFORMS J. Comput. 15 (4), 333–346.
Von Stackelberg, H., 1952. The Theory of The Market Economy. Oxford University Press.
Web, V.R.P., 2019. Capacitated VRP Instances. Networking and Emerging Optimization. Research Group (NEO), University of Malaga, Spain (accessed 28 March 2020).
Wollmer, R., 1964. Removing arcs from a network. Oper. Res. 12 (6), 934–940.
Yao, Y., Edmunds, T., Papageorgiou, D., Alvarez, R., 2007. Trilevel optimization in power network defense. IEEE Trans. Syst. Man Cy Part C 37 (4), 712–718.