Mathematical Programming formulations for rich vehicle routing problems
Hybrid Optimization Methodologies for Transportation Logistics Management (HybOpt)

Contents

1 Introduction

This report presents the mathematical formulations studied and developed to address the critical aspects and constraints of the optimization problems considered in this project. The primary focus is on vehicle and commodity flow models which are integral to solving these complex problems. Additionally, following polytope studies, various valid inequalities are presented for these formulations. By exploring the combinatorial properties of the associated optimization problems and their solution polytopes, this work lays the foundation for the matheuristic algorithms. These studies aim to identify decomposition properties and tractable master-slave schemes, with minimal information loss, that can be efficiently exploited by the proposed algorithms.

Section 2 presents a mathematical formulation for the Set Orienteering Problem (SOP), followed by a proposed subproblem formulation designed to assist local search as an extended operator. Next, a novel two-commodity flow formulation for the Cyclic Production Routing Problem (CPRP) is introduced (Section 3), along with a set of valid inequalities. Subsequently, Section 4 presents formulation for the Production Routing Problem (PRP) is provided, alongside a relaxation subproblem and two additional subproblems that have been instrumental in guiding a matheuristic algorithm to high-quality solutions. Following this, in Section 5, we propose a formulation for the Multivehicle Set Orienteering Problem (MSOP) with tightened valid inequalities. For the Crowdsourced Humanitarian Relief Vehicle Routing Problem, an Integer Programming (IP) model is developed, which incorporates multiple cluster insertions and deletions into a given solution (Section 6). Finally, Section 7 conclude with a Constraint Programming (CP) based formulation for solving the Cumulative Capacitated Vehicle Routing Problem with Time Windows (CCVRPTW).

2 The Set Orienteering Problem

The Set Orienteering Problem generalizes the Orienteering Problem (OP) by considering customers to be divided into mutually exclusive clusters. The profit associated with each cluster is collected by visiting at least one of the customers belonging to this cluster. The problem calls for the determination of the closed route that maximizes the collected profit without violating a given maximum route duration.

SOP was introduced by Archetti et al. (2018) together with a mathematical formulation and a matheuristic algorithm based on tabu search and a mixed integer programming (MIP) based move for performing broad solution modifications. Later, Pěnička et al. (2019) proposed a novel ILP formulation of the problem and developed a Variable Neighborhood Search method applying path and set relocations and exchanges. The most recent research work on SOP is published by Carrabs (2020). It presents a Biased Random-Key Genetic Algorithm (BRKGA) which makes use of three local search procedures to improve the fitness of the solution chromosomes. The solution chromosome is an array of random keys, with each key referring to a customer set. To decode a solution from a given array, the keys (sets) are firstly sorted. The sets whose values are lower than 0.5 are discarded. Then, from the resulting sequence of sets, the node sequence is determined by solving a suitably defined shortest path.

2.1 The Set orienteering problem formulation

The Set Orienteering Problem considers a depot and a set of clustered customers. A vehicle is used to visit customers and collect the associated profit. The goal is to maximize the profit, while respecting the maximum tour length limit. On this basis, let a directed graph $G = (V, A)$, where $V = \{0\} \cup V_c$. Node 0 is the depot, whereas $V_c = \{1, 2, \ldots, n\}$ denotes the set of customer nodes.

Set $A = \{(i,j) : i,j \in V,\ i \neq j\}$ contains arcs that connect all node pairs. The cost c_{ij} required to traverse each arc $(i,j) \in A$ is given. The cost matrix is assumed to satisfy the triangular inequality. The customer set is divided into $|P|$ clusters (sets) $C_g \in P$, where $g = 1, 2, \ldots, l$. Note that clusters are mutually exclusive, i.e., each customer is included in exactly one cluster ($\cup_{g=1}^{l} C_g = C$ and $C_g \cap C_h = \emptyset$, $\forall C_g, C_h \in P, g \neq h$). A profit p_g is associated with every set $C_g \in P$. This profit is collected, if at least one customer $i \in C_g$ is visited by the vehicle. The vehicle must start from the depot and return to it after the tour completion. The total cost (time) of the vehicle tour cannot exceed maximum duration T_{max}. The profit of each cluster may be collected at most once. The objective is to minimize the total collected profit.

In the following, we introduce the notation used in the mathematical formulation of the SOP model. Let y_i be a binary variable equal to 1, iff node $i \in V$ is served. In addition, routing variable x_{ij} is equal to 1, if arc $(i,j) \in A$ is traversed and 0 otherwise. Finally, binary variable z_g is equal to 1, iff the profit of cluster C_g is collected, i.e., any node $i \in C_g$ is visited. In addition, $\delta^+(W) = \{(i,j) \in A : i \in W, j \notin W\}$ and $\delta^-(W) = \{(i,j) \in A : i \notin W, j \in W\}$ for any subset of nodes $W \subset V$. Therefore, $\delta^+(i)$ and $\delta^-(i)$ represent the outgoing and incoming edges of node i, respectively.

Below, the described SOP formulation is provided:

$$\max_{x,y,z} f = \sum_{C_g \in P} p_g z_g \tag{1}$$

subject to

$$\sum_{(i,j) \in \delta^+(i)} x_{ij} = y_i \qquad i \in V \tag{2}$$

$$\sum_{(j,i) \in \delta^-(i)} x_{ij} = y_i \qquad i \in V \tag{3}$$

$$\sum_{(i,j) \in \delta^+(W)} x_{ij} \geq y_h \qquad W \subseteq V \setminus 0, h \in W \tag{4}$$

$$\sum_{(i,j) \in A} c_{ij} x_{ij} \leq T_{max} \tag{5}$$

$$z_g \leq \sum_{i \in C_g} y_i \qquad C_g \in P \tag{6}$$

$$y_i \in \{0,1\} \qquad i \in V \tag{7}$$

$$z_g \in \{0,1\} \qquad C_g \in P \tag{8}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in A \tag{9}$$

The objective function (1) calls for the maximization of the collected profit. Constraints (2) and (3) dictate that a node i is visited, only if exactly one arc enters and exactly one arc leaves the node. Constraints (4) eliminate subtours. These constraints are exponential in number to the size of the problem, thus they need to be generated dynamically as cuts during branching. The total tour cost is limited by constraint (5). Constraints (6) ensure that the profit of each cluster is collected, if and only if at least one vertex of this cluster is visited. Finally, constraints (7)–(9) define the binary variables used in the SOP model. The presented formulation may be directly solved by an MIP solver, if equation (4) is replaced by the MTZ subtour elimination constraints. The interested reader is referred to the paper of Archetti et al. (2018) for more information.

2.2 Decomposition and subproblems formulation

We propose a novel and efficient matheuristic algorithm for solving the Set Orienteering Problem (SOP), combining heuristic and exact methodologies. The approach decomposes the problem into several subproblems, incorporating mathematical programming components within a local search framework. The local search procedure utilizes relocation and swap operators for customers and sets, supported by a tabu-based policy to ensure diversification and prevent cycling. To intensify the search, a Traveling Salesman Problem (TSP) subproblem is periodically solved using the Lin-Kernighan heuristic Helsgaun (2000), alongside a Shortest Path algorithm for optimizing customer sequencing within sets. Additionally, an MIP-based move allows for simultaneous removal and insertion of multiple customer clusters. An Adaptive Memory strategy is employed to enhance exploration, dynamically collecting and recombining solution components from previous iterations, guiding the algorithm towards high-quality solutions.

To enable thorough solution space examination when promising solutions are encountered, we employ an additional operator which allows for broader solution changes that would normally require a possibly high number of local search iterations. This is achieved by solving an IP model which considers multiple cluster insertions and deletions on a given solution S. We call this model Customer Insertion-Deletion IP (*CID*). To ensure that *CID* can be efficiently solved for large-scale instances, all cluster insertion and removal costs are approximated based on the given solution S. The insertion cost of a customer $i \in V_c$ in the route of solution S is the cost of inserting this customer between the best (lower cost increase) pair of nodes. Obviously, only clusters that are not served in S may be inserted, and only clusters already served in S may be removed. The maximum number of cluster insertions and removals are determined by two model parameters. In this way, the problem solver can control i) how drastic the solution modification can be ii) the CPU time required for dealing with the *CID* model, and iii) the quality of the cost approximation. The latter is because when multiple removals and insertions take place simultaneously, the actual cost of the new solution may significantly differ compared to the approximated cost. On the contrary, when a single insertion (deletion) is allowed the approximation cost is the actual cost of performing the insertion (deletion). The idea of an IP model which considers multiple cluster insertions and deletions simultaneously on a given solution is motivated by the work of Manousakis et al. (2022a) who employed a similar idea for the Production Routing Problem.

At this point, it is important to note that the use of approximate costs enhances the *CID* ability to apply drastic modifications. However, this is done in a secure way, because the solution of the *CID* model is directly improved by solving a Shortest Path Problem to determine the best nodes to be served given the updated set of served clusters. Following this, a TSP model is solved to generate the best visit sequence for the selected nodes.

To facilitate the *CID* model description, let us introduce the following notation. Let $\tilde{y}_i$, $\tilde{z}_g$ and $\tilde{x}_{ij}$ be the values of y_i, z_g, and x_{ij} variables, respectively for a given solution S. In addition, let $P^+ = \{C_g \in P \mid \tilde{z}_g = 1\}$ be the set of clusters served in solution S, and $P^- = \{C_g \in P \mid \tilde{z}_g = 0\}$ be the unserved clusters. Also, let constant A_g denote the minimal insertion cost (in terms of c_{ij}) of any customer i of cluster $C_g \in P^-$ (only clusters not visited in the current solution). This minimum insertion cost is obtained by considering all possible insertions of every customer $i \in C_g$ into every possible insertion position of the current solution. Similarly, constants Λ_g are the cost savings of removing cluster $C_g \in P^+$ from solution S. The savings are calculated by connecting the two clusters before and after the removed cluster C_g. Obviously, when only one insertion or removal takes place, the actual cost modification matches the one calculated by the approximation. However, when multiple insertions occur, these can the place in consecutive route positions, and as a result the approximate modification cost does not match the actual

one.

Additionally, let us introduce binary variables α_g and λ_g which are equal to 1, if cluster C_g is inserted or removed, respectively. Variables α_g and λ_g are only defined for clusters such that $\tilde{z}_g = 0$ and $\tilde{z}_g = 1$, respectively. Finally, parameters N_I and N_R limit the maximum number of allowed insertions and removals, respectively.

Below, the *CID* MIP is presented:

$$\max_{\alpha,\lambda} f_2 = \sum_{C_g \in P^-} \alpha_g p_g - \sum_{C_g \in P^+} \lambda_g p_g \tag{10}$$

subject to

$$\sum_{C_g \in P^-} \alpha_g \leq N_I \tag{11}$$

$$\sum_{C_g \in P^+} \lambda_g \leq N_R \tag{12}$$

$$\sum_{(i,j) \in A} c_{ij} \tilde{x}_{ij} + \sum_{C_g \in P^-} \alpha_g A_g - \sum_{C_g \in P^+} \lambda_g \Lambda_g \leq T_{max} \tag{13}$$

$$\alpha_g \in \{0,1\} \qquad C_g \in P^- \tag{14}$$

$$\lambda_g \in \{0,1\} \qquad C_g \in P^+ \tag{15}$$

Objective function (10) maximizes the difference between the additional profit collected by the inserted clusters (first term) minus the profit lost due to cluster removals (second term). Constraint (11) ensures that up to N_I clusters are inserted into the solution. Similarly, constraint (12) limits the number of removed clusters to N_R. Constraint (13) guarantees that the the total solution cost does not exceed the maximum duration T_{max}. The first term represents the current route duration. The second one is the extra cost incurred for serving the inserted clusters and the third one is the cost savings inferred by removing clusters. Note that this constraint does not guarantee that the modified solution cost respects the maximum duration. The use of approximated routing costs may result into infeasible solutions, especially when large values of N_I and N_R are used. Finally, constraints (14) and (15) are the binary variable bounds.

As already mentioned, the optimal *CID* solution may violate the *SOP* maximum duration due to the approximate costs used for cluster removal and insertions. If an infeasible solution is produced, feasibility is heuristically restored by iteratively removing customer clusters. At each iteration, the customer cluster yielding the highest cost savings is removed from the solution. Note that clusters pushed into the solution by the *CID* model are not considered as candidates for being removed when the feasibility is restored.

3 The Cyclic Production Routing Model

The Cyclic Production Routing Problem (CPRP) seeks production, distribution, inventory and routing plans which can be repeated in a cycles of length equal to the studied time horizon. As per the basic PRP model, a product supplier is responsible for replenishing the inventories of customers over a fixed cyclic planning horizon which consists of a predetermined set of periods and guarantees that no stock-outs occur. The supplier decides the time and the quantity of

production and distribution, as well as the routes serving the customers. The differentiating factor of the introduced CPRP variant is the requirement of equal ending and starting inventories for both the production facility and the customers. As the model is clearly affected by the initial inventory levels, the starting inventories of both the customers and the production facility are considered to be decision variables. Unlike, the work of Manousakis (2021) on a periodic PRP which considers starting inventories as parameters, handling initial inventories as variables offers flexibility and increased cost savings. The cost of preparing and establishing the optimal inventory levels is considered negligible. If this is not the case, this preparation cost may be reflected in the objective function. The two-commodity flow PRP model of Manousakis et al. (2022b) is adapted to the examined CPRP. It is worth mentioning that this type of modeling has shown promising results for the IRP which can be regarded as a special case of the basic PRP (Manousakis et al., 2021a).

Let $G = (V, E)$ be an undirected graph where $V = \{0, 1, \ldots, n, n+1\}$ is the set of nodes and $E = \{(i, j) : i, j \in V,\ i < j\}$ is set of edges. Node 0 represents the production facility and node $n + 1$ represents a production facility clone, where the inverse flow originates from. The set of customers is denoted as $V^C = \{1, 2, \ldots, n\}$. The considered planning horizon is considered to repeat in cycles. Each cycle consists of $|T|$ periods (so that the plans are repeated every $|T|$ periods). Let $T = \{1, 2, \ldots, |T|\}$ denote a complete cycle of the planning horizon. To facilitate the model description, let $prev(t)$ denote the index of the period before period t. This is $prev(t) = t - 1$, if $t > 1$. For the first period of the cycle ($t = 1$), $prev(1) = |T|$.

Each node $i \in V$ (both the production facilities and the customers) incurs a unit holding cost h_i for every period $t \in T$. Customer $i \in V^C$ faces a per period $t \in T$ demand d_{ti} and has a limited maximum inventory capacity equal to L_i. At the start of period $t \in T$ the production facility may choose to produce any non-negative quantity p_t up to the production capacity limit C. At any period $t \in T$, if $p_t > 0$, then a production set up cost s_t is incurred. In addition, a per product unit production cost of u_t also applies. The produced quantity may be used directly at the same period t to satisfy customer demands. Each edge $(i, j) \in E$ is associated with a non-negative travel cost c_{ij} that represents the cost of a vehicle for traversing this edge. The supplier delivers any non-negative quantity to every customer $i \in V^C$ at $t \in T$ such that no stock-outs occur. For every period, a homogeneous vehicle fleet of $|K|$ vehicles $K = \{1, 2, \ldots, |K|\}$ each of capacity Q is available at the depot.

We assume a symmetric transportation cost matrix, i.e., $\forall i, j \in E, c_{ij} = c_{ji}$, that satisfies the triangle inequality. Binary undirected routing variables $x_{tij} = 1$ for $i, j \in V,\ i < j$ take value 1, iff any vehicle $k \in K$ traverses edge (i, j) in any direction at period $t \in T$. Binary variables z_{ti} are equal to 1, iff i is visited by any vehicle at period $t \in T$. Similarly, binary variables y_t are 1, iff production takes place at the production facility during period t. Non-negative continuous flow variables f_{tij} with $i < j$ and f_{tji} with $i > j$ represent the load and the residual capacity of the vehicle travelling from i to j at time $t \in T$, respectively. The quantities produced at each period t are denoted as p_t, whereas q_{ti} is the non-negative product quantity delivered to customer $i \in V^C$ at time $t \in T$. The inventory at the end of period t for every node $i \in V$ is captured by the continuous variable I_{ti}. According to the cyclicity constraint, the end inventories of each node $i \in V$ have to be equal to the starting ones, i.e., $I_{|T|,i} = I_{0i}$. The objective of the CPRP is to minimise the overall fixed and variable production, transportation and inventory holding costs for both the supplier and the customers.

Below, the two-commodity flow formulation for the CPRP with the maximum level (ML)

inventory policy is provided, henceforward referred to as *CPRP*:

$$\underset{x,y,z,f,p,q,I}{\text{minimise}}\ g = \sum_{t\in T}\left(u_t p_t + s_t y_t + \sum_{i\in V} h_i I_{ti} + \sum_{i\in V}\sum_{j\in V,\, i<j} c_{ij} x_{tij}\right) \tag{16}$$

subject to

$$\sum_{j\in V, i>j} x_{tji} + \sum_{j\in V, i<j} x_{tij} = 2z_{ti} \qquad i \in V^C, \quad t \in T \tag{17}$$

$$\sum_{j\in V^C} x_{t0j} \le |K| \qquad t \in T \tag{18}$$

$$\sum_{j\in V^C} x_{t0j} = \sum_{i\in V^C} x_{ti,(n+1)} \qquad t \in T \tag{19}$$

$$f_{tij} + f_{tji} = Q x_{tij} \qquad i, j \in V,\ i < j, \quad t \in T \tag{20}$$

$$\sum_{j\in V, i\neq j} f_{tij} = Q z_{ti} - q_{ti} \qquad i \in V^C, \quad t \in T \tag{21}$$

$$\sum_{j\in V^C} f_{t0j} = \sum_{i\in V^C} q_{ti} \qquad t \in T \tag{22}$$

$$\sum_{i\in V^C} f_{ti,(n+1)} = 0 \qquad t \in T \tag{23}$$

$$p_t \le C y_t \quad t \in T \tag{24}$$

$$I_{t0} = I_{prev(t),0} + p_t - \sum_{i\in V^C} q_{ti} \qquad t \in T \tag{25}$$

$$I_{ti} = I_{prev(t),i} + q_{ti} - d_{ti} \qquad i \in V^C, \quad t \in T \tag{26}$$

$$q_{ti} \le L_i + d_{ti} - I_{prev(t),i} \qquad i \in V^C, \quad t \in T \tag{27}$$

$$x_{tij} \in \{0,1\} \qquad i, j \in V,\ i < j, \quad t \in T \tag{28}$$

$$y_t \in \{0,1\} \quad t \in T \tag{29}$$

$$z_{ti} \in \{0,1\} \qquad i \in V^C, \quad t \in T \tag{30}$$

$$0 \le f_{tij} \le Q \qquad i, j \in V,\ i \neq j, \quad t \in T \tag{31}$$

$$0 \le q_{ti} \le \min\left\{L_i + d_{ti}, Q\right\} \qquad i \in V^C, \quad t \in T \tag{32}$$

$$0 \le p_t \le C \quad t \in T \tag{33}$$

$$0 \le I_{ti} \le L_i \qquad i \in V, \quad t \in T \tag{34}$$

The objective function (16) represents the sum of the depot (production facility) setup and variable production costs, the inventory holding costs of both the depot and the customers over the planning horizon, and the transportation costs. Note that no inventory holding costs are considered for the depot clone node. Constraints (17) are the degree constraints for the customers, whereas constraints (18) and (19) bound the number of vehicles leaving and returning

to the depot, respectively. Constraints (20) ensure that the total flow (normal and inverse of each edge) is equal to the vehicle capacity, if and only if the edge is traversed by a vehicle. Constraints (21) are the flow continuity constraints that ensure that the total sum of the flows originating from a node are reduced by the delivery quantity absorbed by the node. If any customer $i \in V^C$, is not visited at period t ($z_{ti} = 0$), then constraints (17) and (20) make sure that both the routing and the flow variables to and from this node are zero. As a result, constraints (21) force the delivered quantity to be zero, $q_{ti} = 0$. Thus, the delivered quantity to a customer may be positive, iff a visit is performed at this customer $z_{ti} = 0$. The total product quantity starting from the depot is given by (22). Constraints (23) ensure that no products return to the production facility, or in other words make sure that the total product quantity leaving the depot is delivered to the customers. Constraints (24) allow production to take place, only when the production facility is open. In this case, the produced quantity cannot exceed the total production capacity limit. Constraints (25) and (26) are the inventory flow preservation constraints over the periods of the planning horizon for the depot and the customers, whereas constraints (27) dictate the ML policy. Note that constraints (25) and (26) jointly set the inventory of every node at the end of the horizon equal to the starting inventory, and thus ensure the repeatability of the decisions made for the the next cycle of the planning horizon. Finally, constraints (28)–(34) enforce integrality, as well as lower and upper bounds on the decision variables. The flow variables f_{tij} are defined by (31) for both directions of each edge, because the existence of flows imposes direction to the model.

The proposed flow formulation eliminates subtours by jointly considering constraints (20)–(23). Therefore, there is no need to introduce $2^{|V|}$ additional constraints to implement the classic DFJ sub-tour elimination constraints (i.e., one for each subset of V), or new variables for the alternative MTZ sub-tour elimination constraints. Hence, the proposed formulation can be solved as-is via any off-the-shelf Mixed Integer Linear Programming (MILP) solver. In the case that the initial inventories cannot be set to the desirable levels, the model can be modified to incorporate fixed initial inventories. Assuming a given initial inventory $I_{init,i}$ for each node $i \in V$, the following constraints have to be added to the model:

$$I_{|T|,i} = I_{init,i} \qquad i \in V, \quad t \in T \tag{35}$$

Constraints (35) force the ending inventory of every node to be equal to the desired initial inventory level.

Regarding the maximum allowed delivery quantity of customer i at period t, we distinguish between two cases that have appeared in the production routing problem literature. The common practice, is to allow the delivered quantity q_{ti} to exceed the maximum capacity L_i making sure that the excessive quantity is consumed during this period t, so that the inventory I_{ti} at the end of the period does not exceed maximum capacity L_i. However, other research works, adopt a stricter assumption forcing the delivery quantity $q_{ti} \leq L_i, \forall i \in V^C, t \in T$ (Adulyasak et al., 2014a, 2015; Qiu et al., 2018). In the present work, we consider the more frequent former case. However, the proposed model can be modified to capture the latter case by replacing constraints (27) and (32) with (36) and (37):

$$q_{ti} \leq L_i - I_{prev(t),i} \qquad i \in V^C, \quad t \in T \tag{36}$$

$$0 \leq q_{ti} \leq min\Big\{L_i, Q\Big\} \qquad i \in V^C, \quad t \in T \tag{37}$$

3.1 Valid inequalities

This section presents valid inequalities that can be used to tighten the proposed CPRP formulation. The valid inequalities are adopted from Manousakis et al. (2022b). For the sake of brevity,

only the adapted valid inequalities are presented below, whereas the valid inequalities that are used as-is to tighten the *CPRP* formulation are described in Appendix B.

The following valid inequalities bound the maximum delivery quantities:

$$q_{ti} \leq \min\Big\{L_i + d_{ti}, Q\Big\} z_{ti} \qquad i \in V^C, \quad t \in T \tag{38}$$

Similarly to constraints (32), valid inequalities (38) bound the maximum delivery quantities with respect to the maximum inventory capacity and the vehicle capacity. The difference is that they are multiplied by the visit variables, and thus ensure that the delivery quantity of a customer may be positive, only if this customer is visited. Note that this is also guaranteed by a combination of constraints in the basic formulation (see model description), but preliminary experiments showed that adding these inequalities straightforwardly, contributes to the lower bound of the root node relaxation.

The concept of sub-deliveries initially introduced by Desaulniers et al. (2016) is used to impose a lower bound on the visits per customer for each period. Specifically, Lefever et al. (2018) bound the visits prior to a specific period t with the minimum number of sub-deliveries (according to the customer capacity) that are required to satisfy the demand of period t. For example, a customer with maximum capacity $L_i = 60$ and demand $d_{ti} = 20$, has to be visited at least once during periods $t \in \{3, 4, 5, 6\}$ in order to satisfy the demand for period $t = 6$.

$$\sum_{t' \in P_{it}^-} z_{t'i} \geq 1 \qquad i \in V^C, \quad t \in T \tag{39}$$

The calculation of set P_{it}^- is described in detail in Lefever (2018) and can be directly applied to the examined CPRP model. It is therefore omitted for the sake of brevity.

4 The Production Routing Model

The generic PRP variant examined, describes the situation in which a manufacturer of a product is responsible for production and replenishment of customers inventories over a given time horizon, ensuring that no stock-outs occur. The decision-maker is responsible for deciding: i) the time periods at which the production takes place, ii) the product quantities that are produced, iii) the timing for replenishing each customer inventory, iv) the associated replenishment quantity and v) the routing of all customer services, with respect to the minimization of the total cost of the system.

For the sake of completeness, we provide a new formulation for the PRP which extends the well-performing formulation of Manousakis et al. (2021b) for the similar IRP. The basic model and valid inequalities are given for a complete PRP definition, and they are also employed for modeling individual subproblems tackled in the context of the proposed algorithm.

Let an undirected graph $G = (V, E)$, where $V = \{0, 1, \ldots, n, n+1\}$ is the set of nodes and $E = \{(i, j) : i, j \in V,\ i < j\}$ is the edge set. Node 0 represents the production facility and node $n + 1$ represents the production facility clone, where the inverse flow originates from. The set $V_c = \{1, 2, \ldots, n\}$ is used to denote the customers. The problem spans over a set of time periods $T = \{1, 2, \ldots, |T|\}$. Each node $i \in V$ starts with initial inventory I_i^0 at period 0 and incurs a unit holding cost h_i for every period $t \in T$. Customer $i \in V_c$ faces a per period $t \in T$ demand d_i^t and has a limited maximum inventory capacity L_i. At the start of period $t \in T$, the production facility (depot) may produce any non-negative quantity p^t up to the per period production capacity limit C, whereas the end inventory I_0^t cannot exceed the production

facility storage capacity L_0. The production facility faces a setup cost s^t for each period that any non-zero quantity is produced, whereas a unit production cost of u^t also applies for each unit of production. The produced quantity may be directly used to satisfy customer demand at the same period t. Each edge $(i,j) \in E$ is associated with a non-negative travel cost c_{ij} that represents the cost for traversing this edge. The supplier delivers any non-negative quantity to each customer $i \in V_c$ at $t \in T$, such that no stock-outs occur. For every period, a homogeneous vehicle fleet of vehicles $K = \{1, 2, \ldots, |K|\}$ each of capacity Q is available at the depot.

We assume a symmetric transportation cost matrix, i.e., $c_{ij} = c_{ji}$. We use binary undirected routing variables $x_{ij}^t = 1$ for $i, j \in V$, $i < j$ iff any vehicle $k \in K$ traverses edge (i,j) in any direction at period $t \in T$. Binary variables z_i^t take a value of 1 iff i is visited by any vehicle at period $t \in T$. Similarly, binary variables y^t take a value of 1 iff production takes place at the production facility during period t. Non-negative continuous flow variables f_{ij}^t and f_{ji}^t represent the load and the residual capacity of the vehicle travelling from i to j at time $t \in T$, respectively. Let p^t be the quantity produced at period t, whereas q_i^t denotes non-negative product quantity delivered to customer $i \in V_c$ at time $t \in T$. Finally, continuous variables I_i^t for each node $i \in V$ represent the inventory at the end of period $t \in T$. The objective of the PRP with the ML inventory policy is to minimize the overall fixed and variable production, transportation and inventory holding costs for both the supplier and the customers.
Below, the two-commodity flow formulation for the PRP with the maximum level (ML) inventory policy is provided, henceforward named *TCF*:

$$\underset{x,y,z,f,p,q,I}{\text{minimize}}\ g = \sum_{t \in T} \left(u^t p^t + s^t y^t + \sum_{i \in V} h_i I_i^t + \sum_{i \in V} \sum_{j \in V,\, i<j} c_{ij} x_{ij}^t \right) \tag{40}$$

subject to

$$\sum_{j \in V, i>j} x_{ji}^t + \sum_{j \in V, i<j} x_{ij}^t = 2z_i^t \qquad i \in V_c \quad t \in T \tag{41}$$

$$\sum_{j \in V_c} x_{0j}^t \leq |K| \qquad t \in T \tag{42}$$

$$\sum_{j \in V_c} x_{0j}^t = \sum_{i \in V_c} x_{i(n+1)}^t \qquad t \in T \tag{43}$$

$$f_{ij}^t + f_{ji}^t = Q x_{ij}^t \qquad i, j \in V,\ i < j \quad t \in T \tag{44}$$

$$\sum_{j \in V, i \neq j} f_{ij}^t = Q z_i^t - q_i^t \qquad i \in V_c \quad t \in T \tag{45}$$

$$\sum_{j \in V_c} f_{0j}^t = \sum_{i \in V_c} q_i^t \qquad t \in T \tag{46}$$

$$\sum_{i \in V_c} f_{i(n+1)}^t = 0 \qquad t \in T \tag{47}$$

$$p^t \leq min\left\{ C, \sum_{i \in V_c} \sum_{t'=t}^{|T|} d_i^{t'} \right\} y^t \quad t \in T \tag{48}$$

$$p^t \leq \sum_{i \in V_c} \left(\sum_{t'=t}^{|T|} d_i^{t'} - I_i^{t-1} \right) - I_0^{t-1} \quad t \in T \tag{49}$$

$$I_0^t = I_0^{t-1} + p^t - \sum_{i \in V_c} q_i^t \qquad t \in T \tag{50}$$

$$I_i^t = I_i^{t-1} + q_i^t - d_i^t \qquad i \in V_c \quad t \in T \tag{51}$$

$$q_i^t \leq L_i + d_i^t - I_i^{t-1} \qquad i \in V_c \quad t \in T \tag{52}$$

$$x_{ij}^t \in \{0,1\} \qquad i,j \in V,\ i < j \quad t \in T \tag{53}$$

$$y^t \in \{0,1\} \quad t \in T \tag{54}$$

$$z_i^t \in \{0,1\} \qquad i \in V_c \quad t \in T \tag{55}$$

$$0 \leq f_{ij}^t \leq Q \qquad i,j \in V,\ i \neq j \quad t \in T \tag{56}$$

$$0 \leq q_i^t \leq min\left\{L_i + d_i^t, Q, \sum_{t'=t}^{|T|} d_i^{t'}\right\} \qquad i \in V_c \quad t \in T \tag{57}$$

$$0 \leq p^t \leq min\left\{C, \sum_{i \in V_c} \sum_{t'=t}^{|T|} d_i^{t'}\right\} \quad t \in T \tag{58}$$

$$0 \leq I_i^t \leq L_i \qquad i \in V \quad t \in T \tag{59}$$

The objective function g represents the sum of the depot setup and unit production costs, the transportation costs and the inventory holding costs of both the depot and the customers over the planning horizon. Note that the holding costs of the initial inventory at period $t = 0$ are also taken into account, whereas no inventory holding costs are considered for the artificial depot node. Constraints (41) are the degree constraints for the customers, while constraints (42) and (43) equate the number of vehicles leaving and returning to the depot and ensure that this number does not exceed $|K|$. Constraints (44) ensure that the total flow (normal and inverse of an edge) is equal to the vehicle capacity if and only if the edge is traversed by a vehicle. Constraints (45) imply the flow as they force the total sum of the flows originating from a node to be reduced by the delivery quantity absorbed by the node. The total product quantity starting from the depot is given by (46). Accordingly, constraints (47) ensure that no products arrive at the depot when routes are terminated, thus guaranteeing that the total product quantity leaving the depot is equal to the quantity delivered to the customers. Constraints (48) ensure that the produced quantity may be positive only when the production facility is open. In this case, the produced quantity cannot exceed the total production capacity or the sum of the remaining demand of all customers. More specifically, according to constraints (49) the produced quantity and the remaining depot inventory cannot exceed the remaining demand after subtracting the existing customer stocks as in any other case the solution would be sub-optimal. Constraints (50) and (51) represent the inventory flow preservation over the periods of the planning horizon for the depot and the customers, whereas constraints (52) implement the ML policy. Finally, constraints (53)–(59) enforce integrality, as well as lower and upper bounds on the decision variables. The flow variables y_{ij}^t are defined by (56) for each edge and direction as the existence of flows imposes direction to the model. Note that period $t = 0$ contributes to the overall holding costs; however, no transportation takes place at this period. It should be underlined that the *TCF* formulation, as a flow model, directly implies the sub-tour elimination from the flow related constraints (44)–(47). Therefore, there is no need to introduce $2^{|V|}$ additional constraints to implement the classic DFJ sub-tour elimination constraints (i.e., one for each subset of V) or new variables for the alternative MTZ sub-tour elimination constraints. In our formulation sub-tours are prevented by imposing the continuous flow of the load and the residual capacity. Hence, it can be solved as-is via any off-the-shelf Mixed Integer Linear Programming (MILP) solver

Regarding the maximum delivery quantity of customer i at period t, we distinguish between two cases that have appeared in the PRP literature. The common practice, is to allow the delivered quantity q_i^t to exceed the maximum capacity L_i making sure that the excessive quantity is consumed during this period t, so that the inventory I_i^t at the end of the period does not exceed maximum capacity L_i. However, other research works, adopt a stricter assumption forcing the delivery quantity $q_i^t \leq L_i, \forall i \in V_c, t \in T$ (Adulyasak et al., 2015; Qiu et al., 2018). In the present work and to enable comparisons, we consider the former case. However, the proposed model can be modified to capture the latter case by replacing constraints (52) and (57) with (60) and (61):

$$q_i^t \leq L_i - I_i^{t-1} \qquad i \in V_c \quad t \in T \tag{60}$$

$$0 \leq q_i^t \leq min\left\{L_i, Q, \sum_{t'=t}^{l} d_i^{t'}\right\} \qquad i \in V_c \quad t \in T \tag{61}$$

Furthermore, in order to facilitate the methodology description, additional notation is introduced for all the algorithmic procedures that refer to a specific solution S. We define $\tilde{y}^t$, $\tilde{z}_i^t$, $\tilde{x}_{ij}^t$, $\tilde{p}^t$ $\tilde{q}_i^t$, $\tilde{I}_i^t$ and $\tilde{f}_{ij}^t$ to be the currently assigned values of y^t, z_i^t, x_{ij}^t, p^t q_i^t, I_i^t and f_{ij}^t in the solution S. Every solution consists of a set of routes R and each route $r \in R$ is assigned to a specific vehicle. Let function $\zeta : (V_c, T) \rightarrow R$, return the route r in which customer $i \in V_c$ is assigned for period $t \in T$. Additionally, let $V_c^r \subseteq V_c$ be the subset of customers that are included in route $r \in R$.

4.1 Production-distribution relaxation

The construction of an initial PRP solution is a challenging task. Common construction heuristics cannot straightforwardly decide the continuous production and delivery quantities which obviously are interconnected with the routing decision level. A simple heuristic design is difficult to ensure feasibility due to the three interconnected decision levels. To overcome this challenge, our proposed heuristic solves a production-distribution relaxation of the proposed *TCF* formulation, named *PD-Rel*. The aim is to generate a partial solution with approximated routing costs. Note that the production setup decisions are crucial for the quality of the final solution, due to the fact that they usually make up for the largest part of the total objective. This means that it is highly unlikely to obtain a good quality solution from a partial solution with poor production schedule characteristics, and vice versa.

The ineffectiveness of modifying the production setup decisions during an iterative improvement procedure is also stated in Adulyasak et al. (2014b). Indeed, preliminary experiments have indicated that modifying the production setup in the context of the hybrid local search causes excessive diversification due to the drastic objective value changes. Therefore, in the proposed scheme, the production setup information defined by *PD-Rel* is maintained throughout the optimization process, whereas the production quantities, the delivery quantities, and distribution plans may be modified.

We adapt the production-distribution relaxation of Adulyasak et al. (2014b) to our two-commodity flow formulation. *PD-Rel* ignores the original routing cost matrix and considers aggregated vehicle capacity over each period. Therefore, the complexity of the problem is drastically reduced, and thus, a high-quality solution of the relaxed problem is usually obtained within limited computational time. Similar to Adulyasak et al. (2014b) for each node $i \in V$, an approximation of the routing (transportation) cost σ_i is calculated as shown in (62)

$$\sigma_i = \min\left\{2c_{0i}, \min_{j,b \in V, j \neq b}\{c_{ij} + c_{ib}\}\right\} \tag{62}$$

The approximation of the cost of visiting a node i, is the minimum between: i) the sum of the distances of the two closest nodes and ii) twice the distance between i and the depot (direct shipping). This is the most optimistic approximation of the visit cost of each node.

The *PD-Rel* formulation is presented below:

$$\underset{y,z,p,q,I}{\text{minimize}}\, \hat{g} = \sum_{t\in T}\left(u^t p^t + s^t y^t + \sum_{i\in V} h_i I_i^t + \sum_{i\in V_c} \sigma_i z_i^t\right) \tag{63}$$

subject to

$$\sum_{i\in V_c} q_i^t \leq |K|Q \qquad t \in T \tag{64}$$

Constraints (48)–(52), (54), (55), (57)–(59).

The objective function (63) is similar to the objective function of the original problem, except for the fact that the last sum involves the approximated and not the actual transportation costs. Constraints (64) ensure that the total delivered quantity for each period t does not exceed the aggregated capacity of all available vehicles. Note that split deliveries are forbidden by the original *TCF* model, thus the aggregated capacity constraint may generate delivery schedules which are *TCF* infeasible. Therefore, it is possible that for a period t, although the aggregated capacity constraint is satisfied for all vehicles together, the delivered quantities cannot be split in a way that the capacity constraint of each individual vehicle is respected. To ensure feasibility, the RHS of inequality (64) is commonly multiplied by a factor between zero and one. The factor is iteratively reduced until a feasible *TCF* production and distribution schedule is constructed (Absi et al., 2015; Chitsaz et al., 2019). However, this is quite restrictive and sacrifices the quality of the solution for the sake of feasibility. Since our algorithm handles infeasible solutions, we have not adopted such a restricting approach.

4.2 Mixed integer linear programming components

The *FILS* incorporates two exact components: i) the *QOPT* which simultaneously optimizes the continuous variables of the delivery, production and inventory quantities, and ii) the *FR* which restores the feasibility of a capacity infeasible solution. Both exact components are presented below.

4.2.1 Quantity optimization method *QOPT*

The quantity optimization method *QOPT* is responsible for optimizing the production and delivery quantities in pursuit of total inventory costs minimization, given the $\tilde{y}^t$, $\tilde{z}_i^t$, $\tilde{x}_{ij}$ and $\tilde{f}_{ij}^t$ values of an incumbent solution S. *QOPT* is responsible for counterbalancing the assumption of fixed decision when a customer delivery quantities are calculated during neighborhood exploration. To do so, it jointly optimizes all delivery and production quantities. The quantity optimization sub-problem tackled by *QOPT* is as follows:

$$\underset{p,q,I}{\text{minimize}}\, g_1 = \sum_{t\in T}\left(u^t p^t + \sum_{i\in V} h_i I_i^t\right) \tag{65}$$

subject to

$$\sum_{i\in V_c^r} q_i^t \leq Q \qquad r \in R \quad t \in T \tag{66}$$

(49)–(52) and the variable bounds (48), (57), (59).

The objective function (65) is made up of the production and inventory terms of the original *TCF* objective function. Note that, routing and setups decisions are not modified by the *QOPT* model. Constraints (66) ensure that the vehicle capacity is not violated. Finally, since y variables are unaffected by the model, the RHS of (48) is a constant stronger variable bound and thus, replaces (58).

4.2.2 Feasibility repair method *FR*

Similarly, to the *QOPT*, the *FR*, optimizes the production and delivery quantities of a given solution S. However, *FR* can also insert, remove and relocate customer visits. In addition, as earlier stated *QOPT* is applied only to feasible solutions, whereas *FR* is applied to both feasible, as well as infeasible solutions. Therefore, the underlying model faced by *FR* extends the *QOPT* sub-problem presented above. To ensure a fast performance, all customer removal and insertions costs are approximated. The number of customer visits that can be added, removed or relocated is bounded, to ensure that: i) the repaired solution is similar to the original one, and ii) the approximation impact on the solution quality is kept to manageable levels. To eliminate capacity infeasibilities, the sub-problem faced by *FR* penalizes vehicle capacity violations. Note that, *FR* not only restores the feasibility of a solution in a near optimal way, but also diversifies the solution by applying several routing modifications, contrary to the *FILS* that can only apply minor routing changes at each iteration. The use of approximation costs instead of the original ones, further enhances the model ability to diversify the search by performing a leap in the solution space. Every repaired solution is further optimized by solving the TSP associated with each route of each period to balance the approximation trade-off.

Let ξ denote the per unit penalty for excess vehicle load. The continuous variable e_r^t denotes the excess load of route r at period t over the actual capacity Q and up to the effective capacity Q^e. Obviously, if any $e_r^t > 0$, route r at period t is infeasible. Also, let constant Δ_i^t denote the minimal insertion cost of customer i in any of the routes of period t. Note that, this is defined even for customers already visited in period t, since the model allows them to be relocated to other routes, or even within the same route. Given the triangular inequalities, the cheapest insertion for any customer cannot be in an empty route. For cases, where the number of vehicles is not limited, this fact does not allow the algorithm to exploit routing alternatives, which involve additional vehicles. In addition, this leads to routes serving many customers which are difficult to be repaired. For these cases, if *FR* cannot restore feasibility, it is executed again with the following modification: the "cheapest" insertion of a customer is set to an empty route with a 25% probability. Let function $\zeta' : (V_c, T) \rightarrow R$ return the route of cheapest insertion of customer i in period t. Similarly, constants Λ_i^t are the savings of removing customer i from period t and are defined only when $\tilde{z}_i^t = 1$.

Additionally, let us introduce binary variables δ_i^t and λ_i^t equal to 1 iff customer i is added or removed from period t, respectively. Variables λ_i^t can be only defined for customers such that $\tilde{z}_i^t = 1$. To avoid non-linear constraints, we introduce delivery quantity variables q'^t_i representing the delivery quantity of relocated customers (moved to another service position of the same period). Therefore, if a customer i is relocated (by being removed and re-inserted), q'^t_i is used instead of q_i^t for the delivery quantity. Finally, parameter a bounds the number of insertions and deletions per route. The value of a depends on the relationship of the objective of the solution to be repaired $Z(S^{rep})$ and the best objective $Z(S^*)$ obtained so far: if $\frac{|Z(S^{rep})-Z(S^*)|}{Z(S^*)} < 0.01$, then $a = a^-$ and otherwise $a = a^+$. This is to control the structural changes caused by the MIP to the solution, depending on the distance of the infeasible solution objective from the best known objective. For conciseness, we denote the constant upper bound of the delivery quantity

to customer i at period t, $\overline{q}_i^t$, as shown in equation (67):

$$\overline{q}_i^t = min\left\{L_i + d_i^t, Q, \sum_{t'=t}^{|T|}(d_i^{t'})\right\} \qquad i \in V_c \quad t \in T \tag{67}$$

Below, the *FR* MIP is presented:

$$\underset{\delta,\lambda,p,q,q',I}{\text{minimize}}\, g_2 = \sum_{t\in T}\left(u^t p^t + \sum_{i\in V} h_i I_i^t + \sum_{r\in R}\xi e_r^t + \sum_{i\in V_c}\Delta_i^t\delta_i^t + \sum_{i\in V_c:\tilde{z}_i^t=1}\Lambda_i^t\lambda_i^t\right) \tag{68}$$

subject to

$$\sum_{i\in V_c^r} q_i^t + \sum_{i\in V_c:\zeta'(i,t)=r,\tilde{z}_i^t=0} q_i^t + \sum_{i\in V_c:\zeta'(i,t)=r,\tilde{z}_i^t=1} {q'}_i^t \le Q^e \qquad r \in R \quad t \in T \tag{69}$$

$$e_r^t \ge \sum_{i\in V_c^r} q_i^t + \sum_{i\in V_c:\zeta'(i,t)=r,\tilde{z}_i^t=0} q_i^t + \sum_{i\in V_c:\zeta'(i,t)=r,\tilde{z}_i^t=1} {q'}_i^t - Q \qquad r \in R \quad t \in T \tag{70}$$

$$\delta_i^t \le \lambda_i^t \qquad i \in V_c \quad t \in T : \tilde{z}_i^t = 1 \tag{71}$$

$$q_i^t \le (1-\lambda_i^t)\overline{q}_i^t \qquad i \in V_c \quad t \in T : \tilde{z}_i^t = 1 \tag{72}$$

$${q'}_i^t \le \delta_i^t \overline{q}_i^t \qquad i \in V_c \quad t \in T : \tilde{z}_i^t = 1 \tag{73}$$

$${q'}_i^t \le \lambda_i^t \overline{q}_i^t \qquad i \in V_c \quad t \in T : \tilde{z}_i^t = 1 \tag{74}$$

$$q_i^t \le \delta_i^t \overline{q}_i^t \qquad i \in V_c \quad t \in T : \tilde{z}_i^t = 0 \tag{75}$$

$$\sum_{i\in V_c:\zeta'(i,t)=r} \delta_i^t \le a \qquad r \in R \quad t \in T \tag{76}$$

$$\sum_{i\in V_c^r} \lambda_i^t \le a \qquad r \in R \quad t \in T \tag{77}$$

$$\delta_i^t \in \{0,1\} \qquad i \in V_c \quad t \in T \tag{78}$$

$$\lambda_i^t \in \{0,1\} \qquad i \in V_c \quad t \in T : \tilde{z}_i^t = 1 \tag{79}$$

$$0 \le e_r^t \le Q^e - Q \qquad r \in R \quad t \in T \tag{80}$$

$$0 \le {q'}_i^t \le \overline{q}_i^t \qquad i \in V_c \quad t \in T : \tilde{z}_i^t = 1 \tag{81}$$

(49)–(52) and variable bounds (48), (57), (59).

Objective function (68) minimizes the total unit production and holding cost, the excessive capacity penalty, as well as the costs for inserting and removing customers. Constraints (69) ensure that the augmented total capacity Q^e (allowing infeasibility) of any vehicle will not be exceeded by the sum of the existing customers delivery quantities (first term), the customers inserted delivery quantities (second term), as well as to customers relocated to the associated route (third term). In a manner similar, constraints (70) sets the augmented vehicle capacity slack variable for excess load. Next, constraints (71) guarantee solution consistency by allowing the addition of an existing customer i in period t iff this customer is removed from its current delivery position (relocation). For existing customers (i.e., $\tilde{z}_i^t = 1$), constraints (72) ensure that $q_i^t = 0$ if customer i is removed. Note that, if customer i is relocated to new delivery position of

the same period t, $q_i^t = 0$ and the delivery quantity associated with the new delivery position is represented by a non-zero ${q'}_i^t = 0$. Constraints (73) and (74) ensure that the relocated delivery quantity ${q'}_i^t$ may be positive only if customer i is both inserted and removed (i.e., relocated). For any customer i inserted in any period t (i.e., $\tilde{z}_i^t = 0$), constraint (75) sets the associated delivery quantity to q_i^t. Inequalities (76) and (77) limit the number of insertions and deletions per route period to a. Additionally, (78)–(81) provide the bounds of the new variables introduced. Note that, the removal, as well as the relocated delivery quantity variables, are defined iff customers are not already routed at the associated period. Finally, since y variables are unaffected by the model, the RHS of (48) is a constant stronger variable bound and thus, replaces (58).

It should be noted that, for the interested reader, a substantially more compact and comprehensible formulation of the aforementioned model is proposed. This model does not make use of the q' variables and thus is non-linear. Preliminary experiments with the non-linear model demonstrated significantly inferior performance compared to the linear one. As mentioned the non linear formulation of the *FR* MIP is more straightforward and comprehensible as there is not need for introducing variables q'. However, it showed inferior performance during preliminary experiments.

Below, we present the non-linear Feasibility Repair *NLFR* MIP:

$$\underset{\delta,\lambda,p,q,I}{\text{minimize}}\, g_2 = \sum_{t\in T}\left(u^t p^t + \sum_{i\in V} h_i I_i^t + \sum_{r\in R}\xi e_r^t + \sum_{i\in V_c}\Delta_i^t\delta_i^t + \sum_{i\in V_c:\tilde{z}_i^t=1}\Lambda_i^t\lambda_i^t\right) \tag{82}$$

subject to

$$\sum_{i\in V_c^r}(1-\lambda_i^t)q_i^t + \sum_{i\in V_c:\zeta'(i,t)=r}\delta_i^t q_i^t \le Q^e \qquad r\in R \quad t\in T \tag{83}$$

$$e_r^t \ge \sum_{i\in V_c^r}(1-\lambda_i^t)q_i^t + \sum_{i\in V_c:\zeta'(i,t)=r}\delta_i^t q_i^t - Q \qquad r\in R \quad t\in T \tag{84}$$

$$\delta_i^t \le \lambda_i^t \qquad i\in V_c \quad t\in T:\tilde{z}_i^t = 1 \tag{85}$$

$$q_i^t \le (1+\delta_i^t-\lambda_i^t)\overline{q}_i^t \qquad i\in V_c \quad t\in T:\tilde{z}_i^t = 1 \tag{86}$$

$$q_i^t \le \delta_i^t\overline{q}_i^t \qquad i\in V_c \quad t\in T:\tilde{z}_i^t = 0 \tag{87}$$

$$\sum_{i\in V_c:\zeta'(i,t)=r}\delta_i^t \le a \qquad r\in R \quad t\in T \tag{88}$$

$$\sum_{i\in V_c^r}\lambda_i^t \le a \qquad r\in R \quad t\in T \tag{89}$$

$$\delta_i^t \in \{0,1\} \qquad i\in V_c \quad t\in T \tag{90}$$

$$\lambda_i^t \in \{0,1\} \qquad i\in V_c \quad t\in T:\tilde{z}_i^t = 1 \tag{91}$$

$$0 \le e_r^t \le Q^e - Q \qquad r\in R \quad t\in T \tag{92}$$

Objective function (82) minimizes the total unit production and holding costs, the excessive capacity penalty and the costs for inserting and removing customers from periods. Constraints (83) ensure that the additional total capacity Q^e (considering the allowed infeasibility) of the vehicles will not be exceeded by the delivery quantities of the existing (first term) and the inserted or relocated customers (second term). Constraints (84) force the slack variable to take the value of excess load. Next, constraints (85) ensure solution consistency by allowing the addition of an existing customer i in period t iff the customer is removed from the current position. For

existing customers (i.e., $\tilde{z}_i^t = 1$), constraints (86) guarantee that quantity may be delivered to an existing customer if the customer is either removed and reinserted or remains in the same position (both δ_i^t and σ_i^t are zero). According, to constraints (87), the delivered quantity of the customers that are not already in the current solution is allowed to be positive only if the customer is inserted. Inequalities (88) and (89) limit the number of insertions and deletions per route period to a. Additionally, (90)–(92) provide the bounds for the new variables introduced. Note that the removal variables are defined iff the customer is not routed is in the incumbent solution.

5 The Multi-Vehicle Set Orienteering Problem

The Multi-vehicle Set Orienteering Problem considers a depot and a set of clustered customers. A vehicle fleet of $|K|$ vehicles is used to visit customers and collect the associated profit. The goal is to maximize the collected profit. Each vehicle $k \in K$ has a maximum tour length limit T_{max}. On this basis, let a directed graph $G = (V, A)$, where $V = \{0\} \cup V_c$. Node 0 is the depot, whereas $V_c = \{1, 2, \ldots, n\}$ denotes the set of customer nodes. Set $A = \{(i, j) : i, j \in V,\ i \neq j\}$ contains arcs that connect all node pairs. The cost c_{ij} required to traverse each arc $(i, j) \in A$ is given. The cost matrix is assumed to satisfy the triangular inequality. The customer set is divided into $|P|$ clusters (sets) $C_g \in P$, where $g = 1, 2, \ldots, l$. Note that clusters are mutually exclusive, i.e., each customer is included in exactly one cluster ($\cup_{g=1}^{l} C_g = C$ and $C_g \cap C_h = \emptyset$, $\forall C_g, C_h \in P, g \neq h$). A profit p_g is associated with every set $C_g \in P$. This profit is collected, if at least one customer $i \in C_g$ is visited by a vehicle. Each vehicle must start from the depot and return to it after the tour completion. The total cost (time) of each vehicle tour cannot exceed maximum duration T_{max}. The profit of each cluster may be collected at most once. The objective is to maximize the total collected profit.

We consider an extended graph of graph G defined as follows. Let $\bar{G} = (\bar{V}, \bar{A})$ where the set of vertices $\bar{V}$ is partitioned as $\{0\} \cup V_c \cup V_d$, where set V_d represents a set of final depots $\{n + 1, \ldots, n + |K|\}$. Depot 0 represents the start depot, and a vehicle route starts at vertex 0, visits vertices from set V_c and ends at a vertex in V_d. The set of arcs $\bar{A}$ is defined as $\bar{A} = \{(0, i) : i \in V_c\} \cup \{(i, j) : i, j \in V_c, i \neq j\} \cup \{(i, j) : i \in V_c, j \in V_d\}$. With each arc $(i, j) \in \bar{A}$ are associated a cost c_{ij} and a travel time t_{ij}. Hereafter, we use (i, j) and a interchangeably to represent an arc in $\bar{A}$.

For any subset $S \subseteq \bar{V}$, the outgoing arcs of set S are denoted $\delta^+(S) = \left\{(i, j) \in \bar{A} : i \in S, j \in \bar{V} \setminus S\right\}$ and similarly $\delta^-(S) = \left\{(i, j) \in \bar{A} : i \in \bar{V}, j \in \bar{V} \setminus S\right\}$ for incoming arcs. If $S = \{i\}$, $\delta^+(i)$ and $\delta^-(i)$ are used for simplicity.

The mathematical formulation uses the following decision variables:

- $\xi_i \in \{0, 1\}, i \in V_c$: equal to one if customer i is visited by a route in solution, zero otherwise.
- $y_a, z_a \geq 0, a \in \bar{A}$: representing the residual and accumulated times of arc (i, j), respectively.
- $x_a \in \{0, 1\}$, $a \in \bar{A}$: equal to one if arc a is selected in solution, zero otherwise.
- $s_g \in \{0, 1\}, g = 1, \ldots, l$: equal to one if cluster g is covered, zero otherwise.

The problem can be formulated as follows:

$$\max \sum_{g=1}^{l} p_g s_g \tag{93}$$

$$\sum_{a\in\delta^+(0)} x_a = |K|, \tag{94}$$

$$\sum_{a\in\delta^+(i)} x_a = \sum_{a\in\delta^-(i)} x_a = \xi_i, \quad \forall i \in V_c \tag{95}$$

$$\sum_{a\in\delta^-(j)} x_a = 1, \quad \forall j \in V_d, \tag{96}$$

$$s_g \leq \sum_{i\in C_g} \xi_i, \quad g = 1,\ldots,l, \tag{97}$$

$$\sum_{a\in\delta^+(i)} z_a = \sum_{a\in\delta^-(i)} z_a + \sum_{a\in\delta^-(i)} t_a x_a, \quad \forall i \in V_c \tag{98}$$

$$\sum_{a\in\delta^-(j)} z_a + \sum_{a\in\delta^-(j)} t_a x_a \leq T_{max}, \quad \forall j \in V_d \tag{99}$$

$$(z_a + y_a) = T_{max}\, x_a, \quad \forall a \in \bar{A}, \tag{100}$$

$$\xi_i \in \{0,1\}, \quad \forall i \in V_c \tag{101}$$

$$y_a, z_a \geq 0, \quad \forall a \in \bar{A} \tag{102}$$

$$x_a \in \{0,1\}, \quad \forall a \in \bar{A} \tag{103}$$

$$s_g \in \{0,1\}, \quad g = 1,\ldots,l \tag{104}$$

The objective function (93) maximises the total profit of the clusters selected in solution. Equation (94) imposes the degree constraint at the vertex 0 whereas constraints (95) state degree constraints for the vertices in V_c visited or selected in solution. Equations (96) impose degree constraints on the set of vertices in V_d. Constraints (97) link variables s and ξ, thus selecting the clusters covered or visited in solution. Equations (98) impose flow constraints for the flow variables z and together with linking equations (100) also on the flow variables y. Constraints (99) imposes that the maximum time duration T_{max} is not exceed by the vehicle routes in solution.

5.1 Valid inequalities

The following inequalities can be used to strengthen the Linear-Programming (LP) relaxation of formulation F which can be written in terms of variables z_a and y_a only using equations (100).

1. Trivial inequalities (TI).
$$z_a + y_a \leq T_{max}, \quad \forall a \in \bar{A}. \tag{105}$$

2. Profit-based (PrB) inequalities.
$$s_g \leq \xi_i, \quad \forall i \in C_g, g = 1,\ldots,l. \tag{106}$$

3. Flow-based (FB) inequalities.
$$y_a \geq t_a \frac{y_a + z_a}{T_{max}}, \quad \forall a \in \bar{A}. \tag{107}$$

4. Path-based (PB) inequalities. Given a solution (y^*, z^*) of the LP-relaxation of formulation F, let x^* be the vector computed as $x^* = (y^* + z^*)/T_{max}$. Consider a path $P = (0, i_1, i_2, \ldots, i_k, j)$ in $\overline{G}$ with $j \in V_d$ and $\{i_1, i_2, \ldots, i_k\} \subseteq V_c$ such that $\sum_{a \in \bar{A}(P)} x^*_a > |\bar{A}(P)| - 1$ with $\sum_{a \in \bar{A}(P)} t_a > T_{max}$, where $\bar{A}(P) \subseteq \bar{A}$ is the set of arcs traversed by path P. The following inequality is valid for formulation F:

$$\sum_{a \in \bar{A}(P)} x_a \leq |\bar{A}(P)| - 1. \tag{108}$$

5. Generalized-subtour-based (GS) inequalities.

$$\sum_{a \in \delta^-(S)} x_a \geq \xi_i, \quad \forall S \subseteq V_c, |S| > 1, i \in V_c, i \in S. \tag{109}$$

6. Time constraints (TC) inequalities. For a given $S \subseteq V_c$, $|S| > 2$, let $T(S)$ be the minimum number of vehicles required to visit customers in S based on the time constraint T_{max}. Then, the following inequality is valid for formulation F:

$$\sum_{a \in \delta^-(S)} x_a \geq T(S). \tag{110}$$

Computing $T(S)$ for the given S is NP-hard, but a lower bound on $T(S)$ can be computed as follows. Let $\bar{t}_i$, $i \in V_c$, be a lower bound on the travelling time to reach customer i in any feasible solution where customer i is visited by a vehicle. Value $\bar{t}_i$ can be computed as $\bar{t}_i = min_{a \in \delta^-(i)}\{t_a\}$. The following inequality is valid:

$$\sum_{a \in \delta^-(S)} x_a \geq \left\lceil \frac{\sum_{i \in S} \bar{t}_i \xi_i}{T_{max}} \right\rceil. \tag{111}$$

Inequalities (111) are nonlinear. A linearised version of the inequalities can be easily derived removing the rounding operator.

7. Knapsack-based (KP) inequalities. The following inequality is valid:

$$\sum_{a \in \bar{A}} t_a x_a \leq |K| T_{max}. \tag{112}$$

Inequality (112) is a knapsack-based inequality where the set of arcs represents the items with corresponding weights (times) and the knapsack capacity is the total maximum travel time of the vehicle fleet.

6 The Crowdsourced Humanitarian Relief Vehicle Routing Problem

To enable thorough solution space examination when promising solutions are encountered and to allow different supply points, we employ an additional operator which allows for broader solution changes that would normally require a possibly high number of local search iterations. This is achieved by solving an IP model which considers multiple cluster insertions and deletions on a given solution S. We call this model Node Insertion-Deletion IP (*NID*). To ensure that

NID can be efficiently solved for large-scale instances, all cluster insertion and removal costs are approximated based on the given solution S. Supply point $i \in V_s$ and customer nodes $i \in V_c$ are allowed to be inserted and removed.

The insertion cost of a node $i \in V_c$ in the route of solution S is the cost of inserting this customer between the best (lower cost increase) pair of nodes. Note that the supply points can only be inserted in the second index of the route (between the depot and the first customer). Obviously, for each route only nodes that are not visited by this route in solution S may be inserted, and only nodes already visited by this route in S may be removed. The maximum number of nodes insertions and removals are determined by two model parameters. In this way, the problem solver can control i) how drastic the solution modification can be ii) the CPU time required for dealing with the *NID* model, and iii) the quality of the cost approximation. The latter is because when multiple removals and insertions take place simultaneously, the actual cost of the new solution may significantly differ compared to the approximated cost. On the contrary, when a single insertion (deletion) is allowed the approximation cost is the actual cost of performing the insertion (deletion). Also, in order to introduce diversification into the solution the model forces a minimum number of supply point removals and insertions. The idea of an IP model which considers multiple node insertions and deletions simultaneously on a given solution is motivated by the work of Manousakis et al. (2022a) who employed a similar idea for the Production Routing Problem.

To facilitate the *NID* model description, let us introduce the following notation. For a given solution S, let $\tilde{w}_i^k$, be the values of w_i^k variables, respectively.

Also, let constant A_i^k denote the minimal insertion cost (in terms of c_{ij}) of node $i \in V_S \cup V_C : \tilde{w}_i^k = 0$. This minimum insertion cost is obtained by considering all possible insertions of every customer $i \in V_C$ into every possible insertion position of the current solution. For nodes $i \in V_S$ the insertion cost is the insertion cost of adding it between the depot and the first node.

Similarly, constants Λ_g are the cost savings of removing a node $i \in V_S \cup V_C : \tilde{w}_i^k = 1$ from route k of solution S. The savings are calculated by connecting the two nodes before and after the removed node i. Obviously, when only one insertion or removal takes place, the actual cost modification matches the one calculated by the approximation. However, when multiple insertions occur, these can the place in consecutive route positions, and as a result the approximate modification cost does not match the actual one.

Additionally, let us introduce binary variables α_i^k and λ_i^k which are equal to 1, if node i is inserted (removed) in (from) route k of solution S, respectively. Variables α_i^k and λ_i^k are only defined for nodes such that $\tilde{w}_i^k = 0$ and $\tilde{w}_i^k = 1$, respectively. Finally, parameters N_C^+ and N_C^- limit the maximum number of allowed insertions and removals, respectively. Parameter N_S is the minimum number of allowed supply point swaps.

Also, let z be the time of visit of the last customer to be served.

Below, the *NID* MIP is presented:

$$\min_{\alpha,\lambda} z \tag{113}$$

subject to

$$\sum_{i \in V_S \cup V_C : \tilde{w}_i^k = 0} \alpha_i^k \leq N_C^+ \qquad k \in K \tag{114}$$

$$\sum_{i \in V_S \cup V_C : \tilde{w}_i^k = 1} \lambda_i^k \leq N_C^- \qquad k \in K \tag{115}$$

$$\sum_{i \in V_C} \tilde{w}_i^k + \sum_{i \in V_C} \alpha_i^k - \sum_{i \in V_C} \lambda_i^k \leq C^{max} \qquad k \in K \tag{116}$$

$$\sum_{k \in k} \alpha_i^k - \lambda_i^k = 0 \qquad i \in V_C \tag{117}$$

$$\sum_{k \in K} \alpha_i^k \geq N_S \qquad i \in V_S \tag{118}$$

$$\sum_{k \in K} \lambda_i^k \geq N_S \qquad i \in V_S \tag{119}$$

$$\sum_{k \in k} \alpha_i^k - \lambda_i^k = 0 \qquad i \in V_S \tag{120}$$

$$\sum_{i \in V} \sum_{j \in V : i \neq j} c_{ij} \tilde{x}_{ij}^k + \sum_{i \in V_S \cup V_C} A_i^k \alpha_i^k - \sum_{i \in V_S \cup V_C} \Lambda_i^k \lambda_i^k \leq z \qquad k \in K \tag{121}$$

$$\alpha_i^k \in \{0, 1\} \qquad i \in V_S \cup V_C : \tilde{w}_i^k = 0 \tag{122}$$

$$\lambda_i^k \in \{0, 1\} \qquad i \in V_S \cup V_C : \tilde{w}_i^k = 1 \tag{123}$$

$$z \in \mathbf{R}_{>0} \tag{124}$$

Objective function (113) minimize the time of service the last customer. Constraint (114) ensures that up to N_C^+ nodes are inserted into the solution. Similarly, constraint (115) limits the number of removed nodes to N_C^-. Constraint (116) guarantees that the the total load does not exceed the maximum capacity C^{max}. The first term represents the current route load. The second one is the extra load incurred for serving the inserted customers and the third one is the load removed by removing customers. Constraints (117) ensure that all customers are visited by not allowing a node to be only removed or only inserted. Constraints (118) and (119) enforce the minimum number of supply point swaps. Constraints (120) ensure that only one supply point is preserved per route. Inequalities (121) force variable z to take the value of the last served node. Of course, due to the approximation, this is an approximation of the actual time. Finally, constraints (122), (123) and (124) are variable bounds.

7 The Cumulative Capacitated Vehicle Routing Problem with Time Windows

The Cumulative Capacitated VRP (CCVRP), is an extension of the Capacitated Vehicle Routing problem. It was first introduced and solved by Ngueveu et al. (2010). The CCVRP shares a similar structure with the CVRP, but differs primarily in its optimization objective. Instead of minimizing the total travel distance, the CCVRP aims to minimize the cumulative sum of customer arrival times. Ngueveu et al. (2010) proposed two memetic algorithms that utilize local search as the primary mechanism for search intensification. Lysgaard and Wøhlk (2014) developed a branch-cut-and-price (BCP) algorithm to solve multiple instances of the CCVRP, while Mattos Ribeiro and Laporte (2012) introduced an adaptive large neighborhood search algorithm. This approach incorporates an adaptive probabilistic model to select the most effective destruction and repair operators. In a comparison study, Ozsoydan and Sipahioglu (2013) evaluated the performance of several optimization algorithms, including genetic algorithms, particle

swarm optimization, and Tabu Search, on the CCVRP. Ke (2018) later proposed a brain storm optimization algorithm, which is capable of solving large-scale instances with up to 1,200 customers. Finally, Kyriakakis et al. (2021) implemented both an ant colony optimization algorithm and a hybrid variable neighborhood descent algorithm to address the CCVRP.

Although the literature on the CCVRP and its extensions is extensive ((Rivera et al., 2015, 2014; Lalla-Ruiz and Voß, 2020)), very few studies have explored the CCVRP with time window constraints (CCVRPTW). To our knowledge, only two works have examined this variation. Liu and Jiang (2019) were the first to introduce and analyze the CCVRPTW, proposing a hybrid large-neighborhood search algorithm that utilizes a constraint relaxation scheme to extend the search space, allowing for the exploration of both feasible and infeasible neighboring solutions. More recently, Kyriakakis et al. (2022) presented a hybrid Tabu Search-variable neighborhood descent algorithm for solving this problem.

Formally, the problem can be defined as follows: Let $G = (V, E)$, be a directed graph, where $V = \{0, 1, \ldots, n\}$ is the set of nodes and $E = \{(i, j) : i, j \in V,\ i < j\}$ is the set of edges, i.e., an edge (i, j) denotes a directed edge between nodes i and j with an associated travel time/distance $t_{i,j}$. A homogeneous fleet of K vehicles is considered, with Q denoting the capacity of each vehicle. Furthermore, a time window $[e_i, l_i]$ and a demand q_i are associated to every node $i \in V$, while a common service time s is required for every visit. Also, let t_i^k define the arrival time of a vehicle k on a node i. Note that, $T_i^k = 0$, when a vehicle k visits node i at the start of the time horizon, or when it does not visit node i. The objective is to minimize the cumulative arrival time of the vehicle to all the customers:

$$Minimize \quad Z = \sum_{k=1}^{K} \sum_{i=0}^{n} T_i^k \tag{125}$$

Note that, when $s = 0$, the objective describes the cumulative distance for visiting all the customers.

7.1 Constraint Programming formulation

A CP model is proposed to solve the problem. Note that, we adopt the IBM CP Optimizer nomenclature, nevertheless, the type of global and local constraints used are available on most of the available CP frameworks of the literature and therefore the model should be straightforward to adapt and reproduce.

Let $v_{i,k}$ an interval variable that denotes a visit of a vehicle k at node i, with a fixed duration equal to s and an interval variable v_i that denotes the visit to i. Also, let c_k and denote the load of a vehicle k. Given the above notation, the CP model is presented below:

$$Z = \sum_{k=1}^{K} \sum_{i=0} StartOf(v_i) \tag{126}$$

$$StartOf(v_i) \geq e_i \quad \forall i \in V \setminus \{0\} \tag{127}$$

$$EndOf(v_i) \leq l_i \quad \forall i \in V \setminus \{0\} \tag{128}$$

$$c_k \leq Q \quad \forall k \in K \tag{129}$$

$$c_k = \sum_{i=1}^{n} PresenceOf(v_{i,k}) d_i \tag{130}$$

$$Alternative(v_i, \{v_{i,k} \quad \forall k \in K\}) \quad \forall i \in V \setminus \{0\} \tag{131}$$

$$NoOverlap(\{v_{i,k} \quad \forall i \in V \setminus \{0\}\}) \quad \forall k \in K \tag{132}$$

Constraint (126) calculates the cumulative arrival time. Constraints (127) and (128) enforce a customer visit to occur only within the associated time window. Note that, Constraints 127 and (128) are satisfied in the case that a vehicle k does not visit a node i. Constraints (129) and (130) ensure that the vehicle capacity constraints are respected. Lastly, Constraints (131) allow for only one vehicle to visit a customer, while Constraints (132) ensure the visits of every vehicle will not overlap with each other. Note that, Constraints (132) takes notice of the edge set E, so as to impose the correct traveling time between two nodes.

References

Absi, N., Archetti, C., Dauzère-Pérès, S., and Feillet, D. (2015). A two-phase iterative heuristic approach for the production routing problem. *Transportation Science*, 49(4):784–795.

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2014a). Formulations and Branch-and-Cut Algorithms for Multivehicle Production and Inventory Routing Problems. *INFORMS Journal on Computing*, 26(1):103–120.

Adulyasak, Y., Cordeau, J. F., and Jans, R. (2014b). Optimization-based adaptive large neighborhood search for the production routing problem. *Transportation Science*, 48(1):20–45.

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2015). The production routing problem: A review of formulations and solution algorithms. *Computers and Operations Research*, 55:141–152.

Archetti, C., Carrabs, F., and Cerulli, R. (2018). The Set Orienteering Problem. *European Journal of Operational Research*, 267(1):264–272.

Carrabs, F. (2020). A biased random-key genetic algorithm for the set orienteering problem. *European Journal of Operational Research*, 292(3):830–854.

Chitsaz, M., Cordeau, J. F., and Jans, R. (2019). A unified decomposition matheuristic for assembly, production, and inventory routing. *INFORMS Journal on Computing*, 31(1):134–152.

Desaulniers, G., Rakke, J. G., and Coelho, L. C. (2016). A branch-price-and-cut algorithm for the inventory-routing problem. *Transportation Science*, 50(3):1060–1076.

Helsgaun, K. (2000). An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130.

Ke, L. (2018). A brain storm optimization approach for the cumulative capacitated vehicle routing problem. *Memetic Computing*, 10(4):411–421.

Kyriakakis, N. A., Marinaki, M., and Marinakis, Y. (2021). A hybrid ant colony optimization-variable neighborhood descent approach for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, 134:105397.

Kyriakakis, N. A., Sevastopoulos, I., Marinaki, M., and Marinakis, Y. (2022). A hybrid tabu search – variable neighborhood descent algorithm for the cumulative capacitated vehicle routing problem with time windows in humanitarian applications. *Computers and Industrial Engineering*, 164.

Lalla-Ruiz, E. and Voß, S. (2020). A popmusic approach for the multi-depot cumulative capacitated vehicle routing problem. *Optimization Letters*, 14(3):671–691.

Lefever, W. (2018). *Stochastic and Robust Optimization Algorithms for the Inventory-Routing Problem and its Extensions*. PhD thesis, Ghent University, Belgium.

Lefever, W., Aghezzaf, E.-H., Hadj-Hamou, K., and Penz, B. (2018). Analysis of an Improved Branch-and-Cut Formulation for the Inventory-Routing Problem with Transshipment. *Computers & Operations Research*, 98:137–148.

Liu, R. and Jiang, Z. (2019). A hybrid large-neighborhood search algorithm for the cumulative capacitated vehicle routing problem with time-window constraints. *Applied Soft Computing*, 80:18–30.

Lysgaard, J. and Wøhlk, S. (2014). A branch-and-cut-and-price algorithm for the cumulative capacitated vehicle routing problem. *European Journal of Operational Research*, 236(3):800–810. Vehicle Routing and Distribution Logistics.

Manousakis, E., Repoussis, P., Zachariadis, E., and Tarantilis, C. (2021a). Improved branch-and-cut for the inventory routing problem based on a two-commodity flow formulation. *European Journal of Operational Research*, 290(3):870–885.

Manousakis, E., Repoussis, P., Zachariadis, E., and Tarantilis, C. (2021b). Improved branch-and-cut for the Inventory Routing Problem based on a two-commodity flow formulation. *European Journal of Operational Research*, 290(3):870–885.

Manousakis, E. G. (2021). *Optimization methods for combined problems in transportation and logistics management*. PhD thesis, Athens University of Economics and Business.

Manousakis, E. G., Kasapidis, G. A., Kiranoudis, C. T., and Zachariadis, E. E. (2022a). An infeasible space exploring matheuristic for the Production Routing Problem. *European Journal of Operational Research*, 298(2):478–495.

Manousakis, E. G., Kasapidis, G. A., Kiranoudis, C. T., and Zachariadis, E. E. (2022b). An infeasible space exploring matheuristic for the production routing problem. *European Journal of Operational Research*, 298(2):478–495.

Mattos Ribeiro, G. and Laporte, G. (2012). An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, 39(3):728–735.

Ngueveu, S. U., Prins, C., and Wolfler Calvo, R. (2010). An effective memetic algorithm for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, 37(11):1877–1885. Metaheuristics for Logistics and Vehicle Routing.

Ozsoydan, F. B. and Sipahioglu, A. (2013). Heuristic solution approaches for the cumulative capacitated vehicle routing problem. *Optimization*, 62(10):1321–1340.

Pěnička, R., Faigl, J., and Saska, M. (2019). Variable Neighborhood Search for the Set Orienteering Problem and its application to other Orienteering Problem variants. *European Journal of Operational Research*, 276(3):816–825.

Qiu, Y., Wang, L., Xu, X., Fang, X., and Pardalos, P. M. (2018). A variable neighborhood search heuristic algorithm for production routing problems. *Applied Soft Computing Journal*, 66:311–318.

Rivera, J. C., Afsar, H. M., and Prins, C. (2014). Multistart evolutionary local search for a disaster relief problem. In Legrand, P., Corsini, M.-M., Hao, J.-K., Monmarché, N., Lutton, E., and Schoenauer, M., editors, *Artificial Evolution*, pages 129–141, Cham. Springer International Publishing.

Rivera, J. C., Afsar, H. M., and Prins, C. (2015). A multistart iterated local search for the multitrip cumulative capacitated vehicle routing problem. *Computational Optimization and Applications*, 61(1):159–187.