CIRRELT

CIRRELT-2024-27

Decomposition Method for a Capacitated Multi-Vehicle Covering Tour Problem with Intermediate Facilities

Vera Fischer
Antoine Legrain
David Schindl

July 2024

Bureau de Montréal
Université de Montréal
C.P. 6128, succ. Centre-Ville
Montréal (Québec) H3C 3J7
Tél : 1-514-343-7575
Télécopie : 1-514-343-7121

Bureau de Québec
Université Laval,
2325, rue de la Terrasse
Pavillon Palasis-Prince, local 2415
Québec (Québec) G1V 0A6
Tél : 1-418-656-2073
Télécopie : 1-418-656-2624

Decomposition Method for a Capacitated Multi-Vehicle Covering Tour Problem with Intermediate Facilities

Vera Fischer[1], Antoine Legrain[2,*], David Schindl[3]

1. University of Fribourg, Fribourg, Switzerland
2. Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Mathematical and Industrial Engineering, Polytechnique Montréal
3. Univeristy of Fribourg and Haute École de Gestion of Geneva, Switzerland

Abstract. We consider a waste collection problem with intermediate disposal facilities to accommodate the usage of smaller, but more sustainable vehicles, with less capacity than the traditional waste collection trucks. The optimization problem consists in determining the locations to place the collection points and the routes of a capacitated collection vehicle that visits these locations. We first present a mixed-integer linear programming formulation that exploits the sparsity of the road network. To efficiently solve practical instances, we propose a solution method that decomposes the problem into a set covering problem to select the locations and a capacitated vehicle routing problem with intermediate facilities to determine the routes and price the set covering solution. We apply column generation to solve this routing problem and present a novel approach for the set covering problem that exploits the properties of the problem to efficiently find a set cover by using a graph theoretical approach. We compare our method with the set construction method and the heuristic Benders decomposition approach presented in a previous work. Computational experiments on instances derived from real-life data confirm the difficulty of our problem and show the superior performance of the developed decomposition method with respect to the number of best solutions and the average gaps to the best solution

Keywords: transit network, bus operations, bus holding, bus line synchronization, real-time control, passenger flow data.

Acknowledgements. The authors gratefully acknowledge the support of Innosuisse under grant 36157.1 IP-EE. They also thank the Hasler Foundation for the support of the research stay in Montreal, Canada and they highly value the involvement of Schwendimann AG in conducting practical experiments and providing the associated data.

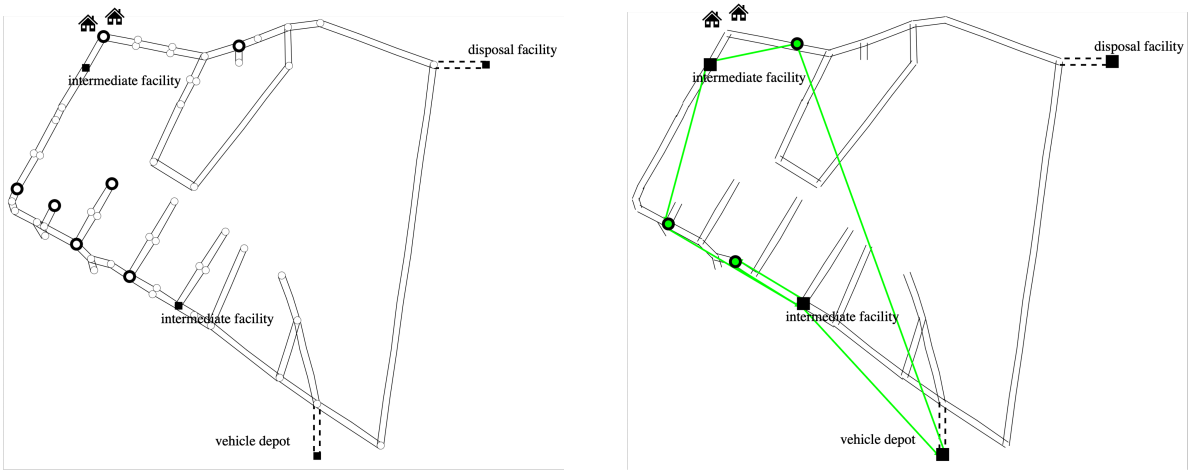
---

* Corresponding author:: antoine.legrain@polymtl.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2024

1. Introduction

In this article, we investigate a waste collection problem involving vehicles that are smaller, but more sustainable than the traditional collection trucks. Such vehicles can be efficient in residential areas with family houses or in city centers with narrow streets. However, due to their limited capacity and driving speed, the installation of intermediate disposal facilities is required to avoid long trips to the final disposal facility. We formulate the problem as a capacitated multi-vehicle covering tour problem with intermediate facilities (Cm-CTP-IF), which is a variation of the capacitated multi-vehicle covering tour problem on a road network (Cm-CTP-R; Fischer et al. 2023b). It was introduced in the context of designing more efficient and sustainable collection systems for non-recoverable waste. Given the road network with a vehicle depot, candidate locations to place collection points and a set of intermediate facilities, the decisions we address in this paper consist in selecting the locations of collection points where residents must leave their bags and determining the routes of a capacitated vehicle. The vehicle performs exactly one rotation that starts and ends at the depot and carries out a sequence of collections with disposals at the available intermediate facilities. For each residential building, a given rank defines and sorts the candidate locations eligible for that building in compliance with some criterion (e.g., walking distance) and we assume that residents will always consider the highest-ranked selected location for leaving their bags. The goal of the Cm-CTP-IF is to determine routes of minimum total travel time that visit a subset of candidate locations, such that all residents are covered (their waste is picked-up at a candidate location from their rank) and the capacity of the vehicle is respected. Figure 1 illustrates the Cm-CTP-IF on a street network of a small part of a Swiss municipality, in which bold nodes represent residential nodes and all nodes (including the bold ones) candidate locations to place collection points.



(a) Street network of a small part of a Swiss municipality. (b) A solution to the Cm-CTP-IF.

Figure 1 Street network of a small part of a Swiss municipality and a possible solution to the Cm-CTP-IF.

The contributions of this paper are the following:

1. We introduce and formulate a new variation of the Cm-CTP-R (Fischer et al. 2023b) that includes intermediate facilities. To handle the large instances derived from real-life data (Fischer et al. 2023b), we develop a solution method that decomposes this problem into the subproblems it is built on: a Set Covering Problem (SCP) and a Capacitated Vehicle Routing Problem with Intermediate Facilities (CVRP-IF). We also propose extensive computational results to demonstrate the advantage of this approach.

2. Taking advantage of the road network structure, we propose a new heuristic to solve the SCP based on the fact that finding a minimum clique cover in a chordal graph is known to be polynomial (Golumbic 2004).

3. We show that a resolution of the Shortest Path Problem with Resource Constraints (SPPRC) with dynamic programming and the concept of *ng*-path (Baldacci et al. 2011) is more efficient on a road network than a complete network for the CVRP-IF.

The remainder of the paper is organized as follows. Section 2 briefly presents research related to our problem. Section 3 formally defines the problem. Section 4 presents the road-network based MILP formulation. Section 5 introduces the general idea of the decomposition method, Section 6 presents the graph theoretical approach for the SCP and Section 7 describes the column generation approach for the CVRP-IF. Section 8 reports the computational experiments, and Section 9 gives some concluding remarks.

2. Related work

The Cm-CTP-R introduced by Fischer et al. (2023b) is closely related to the multi-vehicle covering tour problem (m-CTP; Hachicha et al. 2000), in which up to m vehicle tours are designed that start and end at the depot with minimum total length such that the nodes to cover lie within a preset distance of a tour node (i.e., a node visited by a vehicle). Additional constraints are defined to limit the number of nodes and the length of any tour. The Cm-CTP-R is modeled as a particular version of the m-CTP in which these additional constraints are replaced by vehicle capacity constraints. Moreover, nodes to cover do not only have to be covered by a tour node but must also be allocated to the highest-ranked one that belongs to the solution. Two MILP formulations are proposed, one that exploits the sparsity of the road network and one that is based on a complete-graph representation typically used in vehicle routing problems (VRP; Ben Ticha et al. 2018). Numerical results show that the road-network-based formulation outperforms its customer-based counterpart and provides a more intuitive characterization of the actual network. To solve large instances, a two-phased heuristic approach is developed that addresses the two subproblems the Cm-CTP-R is built on: a SCP to select the candidate locations and a split-delivery vehicle routing problems (SDVRP) to determine the routes. The approach provides good solutions with optimality gaps below 1.7% and finds better solutions for most of the instances that the exact method cannot solve within the time limit. For an overview of relevant research in the context of m-CTP and VRPs with road-network information, we refer the reader to Fischer et al. (2023b) and Ben Ticha et al. (2018), respectively.

Angelelli and Speranza (2002) introduce the term *intermediate facility* as a warehouse where the vehicles can renew completely their capacity, and define a periodic VRP (PVRP) with intermediate facilities which has applications in collection problems. They propose a tabu search algorithm and test different versions of the algorithm on PVRP instances taken from the literature and on randomly generated instances. They show that with an extended set of moves in the algorithm an improvement on the quality of the solutions can be obtained. Markov et al. (2016) consider a complex recyclable waste collection problem that is an extension to the VRP by integrating intermediate facilities, a heterogeneous fixed fleet and a flexible assignment of destination depots. They propose a multiple neighborhood search heuristic which achieves optimality on small instances, shows competitive performance to state-of-the-art solution methods and leads to savings in practice.

The concept of *rotation* is introduced by Crevier et al. (2007) as the set of all routes assigned to a vehicle in the context of a real-life grocery distribution problem in Canada. The authors consider an extension of the multi-depot vehicle routing problem (MDVRP) in which vehicles may be replenished at intermediate depots along their route. They propose a three-phase methodology based on adaptative memory and tabu search for the generation of a set of routes, and on integer programming in the execution of a set partitioning algorithm for the determination of least cost feasible rotations. The method is tested on randomly generated instances and on MDVRP benchmark instances, and reports reasonably fast running times. Tarantilis et al. (2008) propose a three-step algorithmic framework to solve the VRP with intermediate replenishment facilities based on a cost-saving construction heuristic, a tabu search within the variable neighborhood search methodology and a guided local search to eliminate low-quality features from the final solution produced. They are able to produce six new best solutions and reach the best known solutions for all six of the remaining benchmark instances of Crevier et al. (2007). Muter et al. (2014) develop a branch-and-price algorithm for the MDVRP with inter-depot routes (MDVRPI), where vehicles can discharge their loads at intermediate depots. They propose two column generation algorithms, a traditional one-level column generation scheme, and a two-level decomposition scheme in which the pricing problem is solved in two phases and represented with a depot graph consisting of the depot nodes and the routes as arcs to generate rotations in the second phase. They are able to solve exactly some instances with up to 50 customers by applying improvements in the two-phased algorithm. A generic MDVRP in which vehicles are based at multiple depots is studied in Ramos et al. (2020). They propose a new formulation based on the two commodity flow formulation in which the location of the available vehicle fleet and the role of each facility in the network (depot, intermediate facility, or both) are also decisions made by the model. They develop a matheuristic approach to allow the solution of real instances and are able to find some new and better solutions to instances introduced in Muter et al. (2014) and to solve instances introduced in Crevier et al. (2007) for the first time as a MDVRPI.

A combination of the CTP and the multi-depot VRP is introduced by Allahyari et al. (2015), in which a set of vehicle routes over a subset of available customers with the minimum routing and allocation costs is

built, such that the unvisited customers have to be within a predetermined distance of at least one visited customer. They propose a hybrid solution approach combining GRASP, iterated local search and simulated annealing, and show the effectiveness of the proposed solution technique on instances of the introduced problem and three variants of the VRP. Nedjati et al. (2017) consider an extension to the location-routing problem (LRP) with service time restriction, replenishment at intermediate depots, and customer mobility in a predefined walking distance. They present a bi-objective integer linear programming model and propose a heuristic solution method based on the genetic algorithm to solve the different size test problems. They show the good performance of the algorithm and present the impact of valid inequalities in fastening the exact solution procedure.

3. Problem definition

In this section, we formally define the Cm-CTP-IF using the notation and the modeling assumptions introduced in Fischer et al. (2023b). We are given a directed strongly connected graph $G = (V \cup W, A)$ with two node sets V and W, and an arc set A representing the road network. W is a set of nodes with positive demand and V includes nodes that represent candidate locations, intermediate facilities and road intersections. For each demand node $i \in W$, its demand d_i must be satisfied at exactly one node from its rank $V_i^{\text{rank}} \subseteq V$, which contains candidate locations that are within a maximum walking distance from i. We assume that V_i^{rank} is totally ordered based on the assumed criterion to sort candidate locations, which in our case is in increasing order of walking distance. We assume that d_i must be satisfied at the first node in V_i^{rank} at which a vehicle stops. This assumption is based on real-world observations that a resident will simply go to their closest location where a collection point is placed rather than abide by specific assignment decisions. For two nodes $j, j' \in V_i^{\text{rank}}$, $\text{rank}(i, j') < \text{rank}(i, j)$ indicates that node j' is preferred over node j by demand node i, with $\text{rank}(i, j)$ being the index of node j in V_i^{rank}. Then, the set of candidate locations that can be visited by the vehicle are defined as $V^{\text{sto}} = \cup_{i \in W} V_i^{\text{rank}}$.

The set $V^{\text{fac}} \subset V$ contains the available intermediate facilities where the vehicle can dump the load. Note that some of them might not be used in a solution. Let $c_{ll'}$ be the non-negative length of arc $(l, l') \in A$ representing the travel time of (l, l') that can be asymmetric, i.e., $c_{ll'} \neq c_{l'l}$. To capture the time needed to dump the load, a fixed cost is added to each arc entering an intermediate facility $h \in V^{\text{fac}}$. We consider a single vehicle that is located at a distinguished depot node $\sigma \in V$ and is characterized by its capacity Q. It performs exactly one rotation that starts and ends at the depot and is composed of a set of single-facility (i.e., starting and ending at the same intermediate facility) and inter-facility (i.e., connecting two different intermediate facilities) routes with a mandatory visit to a final intermediate facility $h \in V^{\text{fac}}$ before going back to the depot (Crevier et al. (2007)). In contrast to Fischer et al. (2023b), we do not allow for split collection because it is assumed that intermediate facilities are closer to the residential buildings and therefore, overall costs are not significantly improved by this approach. To this end, we assume $d_i \leq Q, \forall i \in W$.

A solution of the Cm-CTP-IF is specified by a sequence of the selected candidate locations $V^{\text{sel}} \subseteq V^{\text{sto}}$ and the given intermediate facilities (i.e., which are not necessarily distinct), and it covers a demand node $i \in W$ if $V_i^{\text{rank}} \cap V^{\text{sel}} \neq \phi$. A solution is feasible if V^{sel} covers all demand nodes (i.e., V^{sel} is a set cover of W) and the vehicle performs exactly one connected rotation collecting the demands d_i at the highest-ranked nodes in V_i^{rank} $\forall i \in W$, while respecting its capacity Q in all routes. The objective of the Cm-CTP-IF is to find a feasible solution with minimum total travel time which is computed as the sum of the lengths of the shortest paths between consecutive nodes in the sequence plus a penalty t^{sto} for each stop performed by the vehicle.

4. Road-network-based formulation

We extend the road-network-based formulation presented in Fischer et al. (2023b) in which decision variables are introduced for each road segment (Letchford et al. (2013)). Let $\mathcal{M} = \{1, \ldots, m\}$ be the set of the m routes, where m defines an upper bound on the number of routes that can be performed based on the vehicle's capacity Q. Let $V^{\text{fac}\sigma} = V^{\text{fac}} \cup \{\sigma\}$ bet the set of available intermediate facilities and the depot. Let $x_{ll'k}$ be an integer variable indicating the number of traversals on arc $(l, l') \in A$ in route $k \in \mathcal{M}$ and y_j be a binary variable taking value 1 if the vehicle stops at node $j \in V^{\text{sto}}$, 0 otherwise. Two non-negative continuous variables q_j and q_{jk} define the total amount of demand satisfied at node $j \in V^{\text{sto}}$ and the amount collected in each route k at node j, respectively, and a binary variable z_{ij} indicates if the demand of $i \in W$ is satisfied at $j \in V_i^{\text{rank}}$. Finally, we introduce a non-negative continuous variable $f_{ll'k}$ to capture the flow passing through arc $(l, l') \in A$ in route $k \in \mathcal{M}$, and a non-negative continuous variable $g_{hh'}$ to indicate the flow between each pair $\{h, h'\} : h, h' \in V^{\text{fac}\sigma}$.

$$\min \quad \sum_{k\in\mathcal{M}}\sum_{(l,l')\in A} c_{ll'}x_{ll'k} + \sum_{j\in V^{\text{sto}}} t^{\text{sto}}y_j \tag{1a}$$

$$\text{s.t.} \quad \sum_{j\in V_i^{\text{rank}}} z_{ij} = 1 \qquad \forall i \in W \tag{1b}$$

$$\sum_{\substack{j'\in V_i^{\text{rank}}:\\ \text{rank}(i,j')>\text{rank}(i,j)}} z_{ij'} \le 1 - y_j \qquad \forall i \in W, j \in V_i^{\text{rank}} \tag{1c}$$

$$\sum_{i\in W: j\in V_i^{\text{rank}}} d_i z_{ij} = q_j \qquad \forall j \in V^{\text{sto}} \tag{1d}$$

$$q_j \le D_j y_j \qquad \forall j \in V^{\text{sto}} \tag{1e}$$

$$y_j \le q_j \qquad \forall j \in V^{\text{sto}} \tag{1f}$$

$$z_{ij} \le y_j \qquad \forall i \in W, j \in V_i^{\text{rank}} \tag{1g}$$

$$\sum_{k\in\mathcal{M}}\sum_{l\in V:(l,j)\in A} x_{ljk} \ge y_j \qquad \forall j \in V^{\text{sto}} \tag{1h}$$

$$\sum_{k\in\mathcal{M}}\sum_{l\in V:(l,h)\in A} x_{lhk} - \sum_{k\in\mathcal{M}}\sum_{l\in V:(h,l)\in A} x_{hlk} = 0 \qquad \forall h \in V^{\text{fac}} \tag{1i}$$

$$\sum_{k\in\mathcal{M}}\sum_{l\in V:(\sigma,l)\in A} x_{\sigma lk} = 1 \tag{1j}$$

$$W^{\text{tot}}x_{ll'k} - f_{ll'k} \ge 0 \qquad \forall (l,l') \in A, k \in \mathcal{M} \tag{1k}$$

$$f_{ll'k} = 0 \qquad \forall (l,l') \in A : \; l \in V^{\text{fac}\sigma}, k \in \mathcal{M} \tag{1l}$$

$$\sum_{h'\in V^{\text{fac}\sigma}} g_{h'h} - \sum_{h'\in V^{\text{fac}\sigma}} g_{hh'} + \sum_{k\in\mathcal{M}}\sum_{l\in V:(l,h)\in A} f_{lhk} = 0 \qquad \forall h \in V^{\text{fac}} \tag{1m}$$

$$\sum_{h\in V^{\text{fac}}} g_{\sigma h} = 0 \tag{1n}$$

$$\sum_{h\in V^{\text{fac}}} g_{h\sigma} = W^{\text{tot}} \tag{1o}$$

$$\sum_{k\in\mathcal{M}} q_{jk} = q_j \qquad \forall j \in V^{\text{sto}} \tag{1p}$$

$$\sum_{j\in V^{\text{sto}}} q_{jk} \le Q \qquad \forall k \in \mathcal{M} \tag{1q}$$

$$\sum_{l'\in V:(l',l)\in A} x_{l'lk} - \sum_{l'\in V:(l,l')\in A} x_{ll'k} = 0 \qquad \forall l \in V \setminus V^{\text{fac}}, k \in \mathcal{M} \tag{1r}$$

$$\sum_{h\in V^{\text{fac}}}\sum_{l\in V:(h,l)\in A} x_{hlk} \le 1 \qquad \forall k \in \mathcal{M} \tag{1s}$$

$$\sum_{l'\in V:(l,l')\in A} f_{ll'k} - \sum_{l'\in V:(l',l)\in A} f_{l'lk} = \begin{cases} q_{lk} & \forall l \in V^{\text{sto}} \\ 0 & \forall l \in V \setminus (V^{\text{sto}} \cup V^{\text{fac}\sigma}) \end{cases} \qquad \forall k \in \mathcal{M} \tag{1t}$$

$$y_j \in \{0,1\}, q_j \ge 0 \qquad \forall j \in V^{\text{sto}} \tag{1u}$$

$$z_{ij} \ge 0 \qquad \forall i \in W, j \in V_i^{\text{rank}} \tag{1v}$$

$$q_{jk} \ge 0 \qquad \forall j \in V^{\text{sto}}, k \in \mathcal{M} \tag{1w}$$

$$x_{ll'k} \in \mathbb{Z}_{\ge 0}, f_{ll'k} \ge 0 \qquad \forall (l,l') \in A, k \in \mathcal{M} \tag{1x}$$

$$g_{hh'} \ge 0 \qquad \forall h, h' \in V^{\text{fac}\sigma} \tag{1y}$$

The objective function (1a) expresses the total cost, which is computed as the sum of the total travel times plus t^{sto} times the total number of stops. Constraints (1b) ensure that the demand of node $i \in W$ is satisfied at exactly one candidate location from its rank. Constraints (1c) state that this demand is satisfied at the first node in V_i^{rank} at which the vehicle stops. Constraints (1d) guarantee that the total demand that is assigned to node $j \in V^{\text{sto}}$ is equal to the total amount that is collected at that node. Constraints (1e) link the variables q_j with the variables y_j and impose an upper bound $D_j = \min\{(\sum_{i \in W : j \in V_i^{\text{rank}}} d_i), Q\}\ \forall j \in V^{\text{sto}}$ on the total amount that can be satisfied at node j. Constraints (1f) link the same variables in the opposite direction such that the vehicle can stop at a node j only if some demand is assigned to it. This prevents from suboptimal solutions in which selected nodes are not needed to satisfy demand. Constraints (1g) link the variables z_{ij} with the variables y_j by imposing that demand can only be assigned to a node $j \in V_i^{\text{rank}}\ \forall i \in W$ which is visited by the vehicle. Constraints (1p) link the variables q_j and q_{jk} to each other such that the total demand satisfied at node $j \in V^{\text{sto}}$ corresponds to the total demand satisfied in all routes. Constraints (1q) limit the demand satisfied in route $k \in \mathcal{M}$ to the vehicle's capacity Q.

Constraints (1h) - (1j) and (1r) - (1s) force the variables x to take values that make up a valid rotation. More precisely, constraints (1h) specify that the vehicle can stop at a node $j \in V^{\text{sto}}$ only if it traverses an incoming arc of node j at least once. Constraints (1i) state the degree constraints for each facility $h \in V^{\text{fac}}$. Constraint (1j) ensures that the depot is visited exactly once in a rotation. Constraints (1r) define that the vehicle enters and leaves any node $h \in V \setminus V^{\text{fac}}$ the same number of times in route $k \in \mathcal{M}$. Constraints (1s) ensure that in each route $k \in \mathcal{M}$ at most one facility is visited.

Constraints (1k) - (1n) ensure connectivity by defining the flow on the variables $f_{ll'k}$ and $g_{hh'}$. Constraints (1k) link the variables $f_{ll'k}$ with the variables $x_{ll'k}$. If $x_{ll'k} = 0$, *i.e.*arc (l, l') is not traversed in route k, then the flow $f_{ll'k}$ does not pass through it, and therefore $f_{ll'k} = 0$. If $x_{ll'k} = 1$, then $W^{\text{tot}} = \sum_{i \in W} d_i$ defines an upper bound on the total flow. Constraints (1l) impose a 0 outflow for $l \in V^{\text{fac}\sigma}$. Constraints (1m) link the variables $f_{ll'k}$ with the variables $g_{hh'}$ by passing the flow on the route level f to the rotation level g. Constraint (1n) defines a 0 outflow at the depot. Constraint (1o) enforces that the total amount going into the depot is equals the total amount satisfied in the rotation. To ensure a last visit to an intermediate facility before going back to the depot, arcs entering the depot are defined in G only for pairs $\{l, \sigma\} : l \in V^{\text{fac}}$. Constraints (1t) define that the net outflow of any node $l \in V^{\text{sto}}$ must be the amount q_{lk} satisfied at node l in route $k \in \mathcal{M}$. For any other node, these constraints impose a 0 net outflow. Finally, constraints (1u) - (1y) define the domain of the decision variables. Note that the binary constraints on variables z_{ij} are unnecessary as automatically enforced.

5. Decomposition method

To solve large instances derived from real-life data, we propose a solution method that decomposes the Cm-CTP-IF into a SCP and a CVRP-IF. In our context, the SCP has a particular structure that can be exploited.

We do so by introducing a novel approach for the SCP which consists in finding a *minimum clique cover* in an intersection graph of the demand nodes' walkable subgraphs (Section 6). This method returns a set cover V^{sel} defining an instance of CVRP-IF, which is tackled by a column generation technique (Section 7).

Algorithm 1 presents the general procedure of the decomposition method. Line 1 initialises the best solution *bestSol*. Line 3 calls the first phase of the method in which a set cover V^{sel} is created (Section 6). This phase corresponds to the optimization problem (1b) $-$ (1g). If the set cover V^{sel} has not been treated by the second phase, we build an integer routing solution (Line 5). The second phase corresponds to (1h) - (1t), which is solved by column generation (Section 7). Finally, Lines 6 - 9 update the best solution. Once the stopping criterion is met (i.e. in the computational experiments we set a 12-hour time limit, see Section 8), we improve the best solution with the diving heuristic (Section 7.4) before returning it.

Algorithm 1 Decomposition method

Input: $G = (V \cup W, A), \quad V_i^{\text{rank}} \ \forall i \in W$

Output: Solution to the Cm-CTP-IF

1: Define bestSol $\leftarrow \emptyset$
2: **while** the stopping criterion is not met **do**
3: Create set cover V^{sel} in the first phase (Section 6)
4: **if** V^{sel} was not already treated by the second phase **then**
5: Generate an integer routing solution S from V^{sel} by column generation (Section 7) and give a feedback to nodes in V^{sel} (Section 7.3)
6: **if** cost(S) $<$ cost(bestSol) **then**
7: bestSol $\leftarrow$ S
8: Improve bestSol with diving heuristic (Section 7.4)
9: **return** bestSol

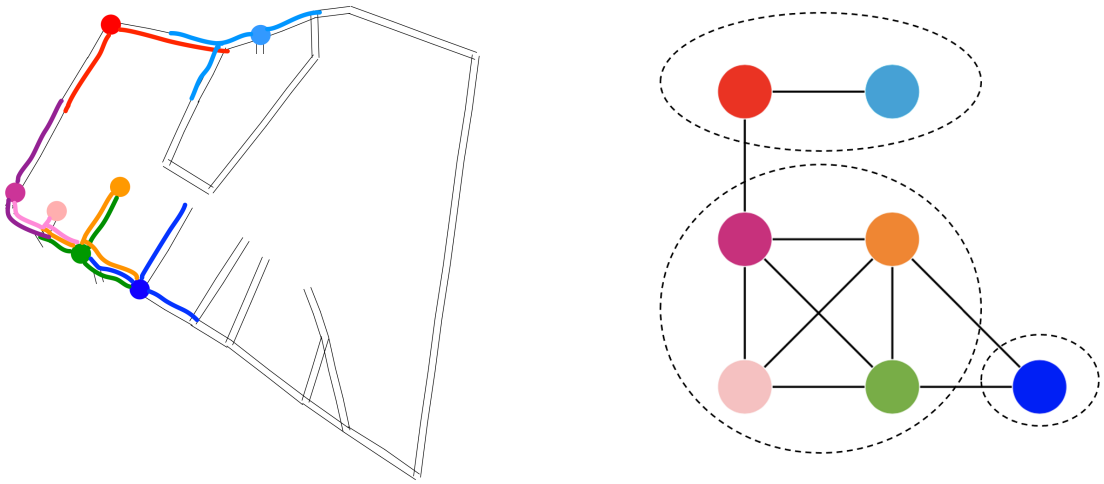
6. Set Covering Problem (SCP)

The SCP is solved with a novel minimum clique cover approach that benefits from the structure of the road network and the properties of the underlying graphs. We refer to this approach as the *Clique* method in the remainder of the paper.

6.1. A minimum clique cover in a chordal graph

For each demand node $i \in W$, consider the set of candidate locations $j \in V_i^{\text{rank}}$. Assume first that each j is at a distance sufficiently small from i, such that there is only one possible path on the road network from i to j. If this is the case, the union over $j \in V_i^{\text{rank}}$ of these paths is a tree T_i. Assume further that taking the union over $i \in W$ of all these trees does not create any cycle, i.e., induces a forest (i.e., a possibly disconnected disjoint union of trees). Notice that this is likely to happen if the road network is relatively

sparse, and the sets V_i^{rank} are defined by a reasonably small maximum walking distance, such as depicted in Figure 2a. Clearly, for any two demand nodes $i, i' \in W$, their trees T_i and T_i' intersect if and only if $V_i^{\text{rank}} \cap V_{i'}^{\text{rank}} \neq \emptyset$. Consider the intersection graph of these trees $G^{\text{I}} = (W, E^{\text{I}})$, such that $\{i, i'\} \in E^{\text{I}}$ if and only if $V_i^{\text{rank}} \cap V_{i'}^{\text{rank}} \neq \emptyset$. Figure 2b displays the intersection graph corresponding to the set of trees of Figure 2a. Under the above assumptions, such a graph has the property of being chordal, or equivalently without chordless cycle on at least 4 nodes. Indeed, as was shown in Gavril (1974), a graph is chordal if and only if it is the intersection graph of a set of subtrees in a tree. Still under the above assumptions, another useful property is that for any clique C in G^{I}, there exists at least one node j which belongs to all the sets V_i^{rank} $\forall i \in C$ (Helly property; Helly 1923). Thus, the problem of finding a minimum set of candidate locations covering all demand nodes is equivalent to finding a minimum set of cliques in G^{I} covering all its nodes. It turns out that in chordal graphs, this problems admits an efficient polynomial algorithm (Golumbic 2004), which we are using here.



(a) Family of subgraphs T_i $\forall i \in W$. (b) A minimum clique cover in the intersection graph G^{I}.

Figure 2 Family of subgraphs T_i $\forall i \in W$ and a minimum clique cover in its intersection graph G^{I}.

6.2. A clique cover heuristic

The above sufficient conditions for G^{I} to be chordal may however in general not be satisfied. To enable us to exploit the problem structure, we first build a reduced graph $G^{\text{D}} = (V^I \cup W, E)$ where $V^I \subset V$ is the set of road intersections and E is a set of undirected edges that reconnects the nodes in G^{D} based on the road network information in G. More precisely, for each node $l \in V \backslash (V^I \cup W)$ with its two incident arcs $(l', l), (l, l'') \in A$, we add an edge $\{l', l''\}$ to E. Notice that a node $l \in V \backslash (V^I \cup W)$ always has exactly two incident arcs, since it will never be placed at the end of a road unless it is a demand node (in W), and only intersection nodes (in V^I) have more than two incident arcs.

We define an edge weight $w_{l'l''}$ $\forall\{l', l''\} \in E$ which is initialized to 0 for every edge and will be updated with $w_e = \alpha\overline{w_e} + (1-\alpha)w_e$ $\forall e \in E$ based on the feedback $\overline{w_e}$ provided by the routing problem (Section 7.3). We then construct a minimum spanning tree (MST) in G^{D} and define $V_i^{\text{MST}} \subseteq V_i^{\text{rank}}$ $\forall i \in W$ so that $j \in V_i^{\text{MST}}$ is within the maximum walking distance from i in MST (instead of G). Let T_i $\forall i \in W$ be the union over $j \in V_i^{\text{MST}}$ of the paths from i to j in MST. Since each tree T_i is a subtree of MST, the resulting intersection graph $G^{\text{I}} = (W, E^{\text{I}})$ with $E^{\text{I}} = \{\{i, i'\} | i \neq i' \text{ and } V_i^{\text{MST}} \cap V_{i'}^{\text{MST}} \neq \emptyset\}$ is chordal.

Algorithm 2 presents the detailed procedure to find such a clique cover in the intersection graph G^{I}. Given $G = (V \cup W, A)$ and ranks V_i^{rank} $\forall i \in W$, we first use the algorithm from Kruskal (1956) to find a MST in the reduced graph $G^{\text{D}} = (V^I \cup W, E)$ and define updated ranks $V_i^{\text{MST}} \subseteq V_i^{\text{rank}}$ $\forall i \in W$ based on the MST. Then we create the intersection graph $G^{\text{I}} = (W, E^{\text{I}})$ as above, and apply the exact algorithm from Golumbic (2004) to find a minimum clique cover in G^{I}.

Algorithm 2 Clique cover

Input: $G = (V \cup W, A), \quad V_i^{\text{rank}} \ \ \forall i \in W$

Output: Clique cover of the intersection graph

1: Find a MST in the reduced graph $G^{\text{D}} = (V^I \cup W, E)$ (see Kruskal 1956) and define $V_i^{\text{MST}} \subseteq V_i^{\text{rank}}$ $\forall i \in W$

2: Construct the intersection graph $G^{\text{I}} = (W, E^{\text{I}})$ with $\{i, i'\} \in E^{\text{I}} \Leftrightarrow V_i^{\text{MST}} \cap V_{i'}^{\text{MST}} \neq \emptyset$

3: Find a minimum clique cover in G^{I} (see Golumbic 2004)

4: **return** Clique cover

6.3. A set cover heuristic

The procedure to generate a set cover V^{sel} from the obtained clique cover is shown in Algorithm 3. For each clique C of the clique cover, we select a candidate location $j \in V_i^{\text{rank}}$ that maximizes the total coverage, i.e., the number of demand nodes that are covered with $V^{\text{sel}} \cup \{j\}$. Given the selected set cover nodes in V^{sel}, we apply two additional moves to repair and improve V^{sel} (Lines 5 - 11 in Algorithm 3). In case and as long as there exists a node $j \in V^{\text{sel}}$ that aggregates more waste than the vehicle's capacity Q, we select a demand node i that is satisfied at node j and add a node $j' \in V_i^{\text{rank}} \setminus V^{\text{sel}}$, such that $\text{pref}(i, j') < \text{pref}(i, j)$, to V^{sel}. Finally, we remove redundant nodes from V^{sel} where a node $j \in V^{\text{sel}}$ is redundant if its total demand with respect to V^{sel}, denoted by $q(V^{\text{sel}})_j$, does not exceed Q. Recall that the total demand $q(V^{\text{sel}})_j$ associated with any collection node $j \in V^{\text{sel}}$ depends on the current composition of the set cover V^{sel}, since the demand d_i $\forall i \in W$ is assigned to the first node in V_i^{rank} that belongs to V^{sel}.

To improve V^{sel} further, we construct a giant tour (also known as traveling salesman tour) with the procedure presented in Fischer et al. (2023b). Such a giant tour starts and ends at the depot and visits nodes in $V^{\text{sel}} \cup V^{\text{alt}}$, where V^{alt} contains alternative nodes to the nodes in V^{sel}. An alternative to $j \in V^{\text{sel}}$ is another node $j' \in V^{\text{sto}} \setminus V^{\text{sel}}$ that can cover at least the same demand nodes and might bring some improvement in

Algorithm 3 Set cover V^{sel}

Input: Clique cover (CC)

Output: V^{sel}

1: Define set cover $V^{\text{sel}} \leftarrow \phi$
2: **for** each clique $C \in$ CC **do**
3: Select node $j \in \cap_{i \in C} V_i^{\text{rank}}$ with maximum coverage
4: $V^{\text{sel}} \leftarrow V^{\text{sel}} \cup \{j\}$
5: **while** $\exists\, j \in V^{\text{sel}}$ with $q(V^{\text{sel}})_j > Q$ **do**
6: Find a demand node i which is satisfied at node j
7: Find a node $j' \in V_i^{\text{rank}} \setminus V^{\text{sel}} : \text{pref}(i, j') < \text{pref}(i, j)$
8: $V^{\text{sel}} \leftarrow V^{\text{sel}} \cup \{j'\}$
9: **for** $j \in V^{\text{sel}}$ **do**
10: **if** $V^{\text{sel}} \backslash \{j\}$ is a set cover and $q(V^{\text{sel}} \backslash \{j\})_{j'} \leq Q \; \forall j' \in V^{\text{sel}} \backslash \{j\}$ **then**
11: $V^{\text{sel}} \leftarrow V^{\text{sel}} \backslash \{j\}$
12: **return** V^{sel}

the routing (e.g., nodes across the street). The construction of the giant tour is based on the savings obtained from the gradual insertion of nodes (inspired by Clarke and Wright 1964), which is repeated until the nodes visited in the tour form a set cover such that V^{sel} can be redefined with these nodes. We refer the reader to Fischer et al. (2023b) for more details on this procedure. Finally, if needed, V^{sel} is repaired according to Lines 5 - 11 in Algorithm 3, such that nodes are not assigned more demand than the vehicle's capacity.

6.4. Diversification

To diversify the generation of set covers and to prevent stalling in the search of a new set cover, we apply the following two mechanisms in Lines 1 and 3 of Algorithm 2. To find a minimum clique cover in the intersection graph G^{I} (Algorithm 2), the applied algorithm (Golumbic 2004) first defines a perfect elimination ordering (PEO) of the nodes in G^{I}. We use the maximum cardinality method (Rose and Tarjan 1975) to define such a PEO, which can be initialized at any node in G^{I}. The first diversification mechanism is to choose a random start node in G^{I} for the maximum cardinality method. The second diversification mechanism revolves around the edge weights to find a MST in the reduced graph G^{D} (Algorithm 2), which are updated based on the feedback given from the routing solution with the function $w_e = \alpha \overline{w_e} + (1-\alpha) w_e \; \forall e \in E$. Following the results of the computational experiments (see Section 8.3), the alpha value is initialized to 1. If the first diversification mechanism and the weight updates are not able to produce a new set cover, we define $\alpha = \alpha - 0.1$ until either a new set cover is found or α reaches the value 0, in which cases we reset α to its initial setting (i.e., $\alpha = 1$).

7. Capacitated Vehicle Routing Problem with Intermediate Facilities (CVRP-IF)

This section describes the column generation approach to address the CVRP-IF in the proposed decomposition method (Section 5). Column generation is a technique to solve problems with a large number of variables by iteratively adding some of the variables to the model (Dantzig and Wolfe 1960). It seems to be a promising strategy for the Cm-CTP-IF since the routes of a small vehicle with intermediate facilities tend to be shorter than for instance the tours considered in Cm-CTP-R. Given a set cover solution $V^{\text{sel}} \subseteq V^{\text{sto}}$ to the SCP, we follow the standard procedure in which iteratively the linear relaxation of a restricted master problem (Section 7.1) is solved followed by a subproblem (Section 7.2) consisting in finding feasible routes of negative reduced cost. This process terminates as soon as no route with negative reduced cost can be found. Finally, we solve a MILP with all generated columns set as binary variables to obtain an integer routing solution. At the end of Algorithm 1 and before returning the best solution *bestSol*, we apply a diving heuristic (Section 7.4) in which additional columns may be generated to improve the integer routing solution of *bestSol*.

7.1. Routing master problem

The compact formulation (1h) $-$ (1t) can be reformulated in the spirit of a Dantzig-Wolfe decomposition (Dantzig and Wolfe 1960). Let $\overline{q}_j \ \forall j \in V^{\text{sto}} : \overline{q}_j = 0 \ \forall j \notin V^{\text{sel}}$ represent a solution to the SCP and define the amount of demand assigned to each node j. Let R be a set of routes starting and ending at the intermediate facilities or the depot $V^{\text{fac}\sigma}$, with $R_h^+ \subseteq R$ and $R_h^- \subseteq R$ representing the subsets of routes that start and end at h, respectively. Note that R is different from $\mathcal{M}$, thus we use index $r \in R$ instead of index $k \in \mathcal{M}$. Let $W_r = \sum_{j \in V^{\text{sel}}} q_{jr}$ be the amount of demand picked up in route $r \in R$, with q_{jr} indicating the amount of demand collected at node j in route r, and let $c_r \ \forall r \in R$ be the travel cost of route r. We introduce a continuous variable x_r indicating if route $r \in R$ is selected and refer to the following formulation as the routing master problem (RMP).

$$\min \quad \sum_{r \in R} c_r x_r + \sum_{j \in V^{\text{sto}}} c^{\text{pick}} \nu_j + c^{\text{coll}} \lambda \tag{2a}$$

$$\text{s.t.} \quad \sum_{r \in R} q_{jr} x_r + \nu_j = \overline{q}_j \qquad \forall j \in V^{\text{sto}} \qquad (\xi_j) \tag{2b}$$

$$\sum_{r \in R_h^-} x_r - \sum_{r \in R_h^+} x_r = 0 \qquad \forall h \in V^{\text{fac}\sigma} \qquad (\beta_h) \tag{2c}$$

$$\sum_{r \in R_\sigma^+} x_r = 1 \qquad (\gamma) \tag{2d}$$

$$W^{\text{tot}} \sum_{r \in R_h^+ \cap R_{h'}^-} x_r - g_{hh'} \geq 0 \qquad \forall h, h' \in V^{\text{fac}\sigma} \qquad (\delta_{hh'}) \tag{2e}$$

$$\sum_{h' \in V^{\text{fac}\sigma}} g_{h'h} - \sum_{h' \in V^{\text{fac}\sigma}} g_{hh'} + \sum_{r \in R_h^-} W_r x_r \geq 0 \qquad \forall h \in V^{\text{fac}} \qquad (\zeta_h) \tag{2f}$$

$$\sum_{h \in V^{\text{fac}}} g_{\sigma h} = 0 \qquad (\eta) \tag{2g}$$

$$\sum_{h \in V^{\text{fac}}} g_{h\sigma} + W^{\text{tot}} \lambda = W^{\text{tot}} \qquad (\kappa) \tag{2h}$$

$$x_r \geq 0 \qquad \forall r \in R \tag{2i}$$

$$\nu_j \geq 0 \qquad \forall j \in V^{\text{sto}} \tag{2j}$$

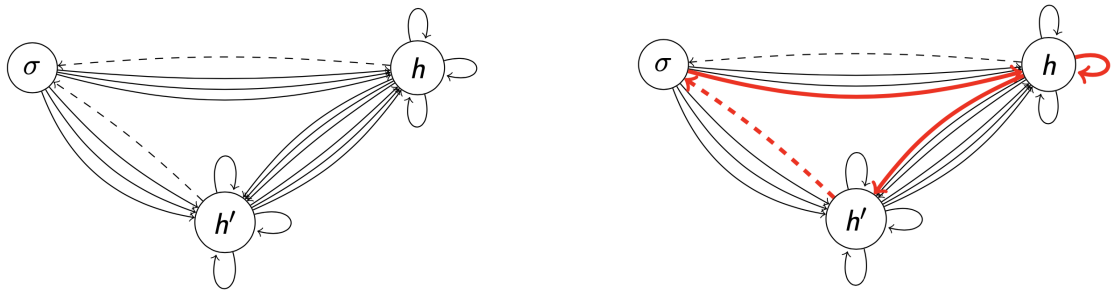
$$\lambda \geq 0 \tag{2k}$$

$$g_{hh'} \geq 0 \qquad \forall h, h' \in V^{\text{fac}\sigma} \tag{2l}$$

We build a multi-digraph $H = (V^{\text{fac}\sigma}, A^{\text{H}})$ where each arc $a \in A^{\text{H}}$ represents a route $r \in R$ starting and ending at the respective nodes. Figure 3 displays such a graph and a possible solution. Note that the vehicle can dump its load at intermediate facilities only, which is why routes ending at σ (i.e., dashed lines) do not visit any set cover nodes. Starting with a set of empty routes connecting each pair of nodes (i.e., no collection), the subproblem then generates new routes for each pair $\{h, h'\}, h \in V^{\text{fac}\sigma}, h' \in V^{\text{fac}}$ which are added to A^{H} if their reduced costs are negative. To ensure feasibility of the problem, artificial variables ν_j for nodes $j \in V^{\text{sto}}$ that are not satisfied and λ for demand that is not collected are introduced and highly penalized in the objective function by c^{pick} and c^{coll}, respectively. The goal of the RMP is to select a subset of routes $R^{V^{\text{sel}}} \subseteq R$ that collect the demand from the set cover nodes V^{sel} at minimum total cost.

7.2. Pricing problem

The routing subproblem, which aims at finding routes with negative reduced cost with respect to the RMP, is also called the pricing problem (PP). The PP formulation (3) makes use of variables analogous to those



(a) $H = (V^{\text{fac}\sigma}, A^{\text{H}})$, with A^{H} representing routes in R.

(b) A possible solution $R^{V^{\text{sel}}} \subseteq R$ in red.

Figure 3 Graph $H = (V^{\text{fac}\sigma}, A^{\text{H}})$ and a possible solution $R^{V^{\text{sel}}}$ to the RMP in red.

defined in the complete formulation (1) and the dual variables of the RMP (2), which are depicted in parentheses next to their constraints. As defined, the PP allows to generate routes that collect any amount of demand at any node with $0 \leq q_j \leq \overline{q}_j \ \forall j \in V^{\text{sto}}$. Let $\mathbb{1}_{\{\sigma\}}(h)$ indicate if the starting node h corresponds to σ.

$$\text{min} \quad \sum_{(l,l')\in A} c_{ll'} x_{ll'} - \sum_{j\in V^{\text{sto}}} (\xi_j + \zeta_{h'}) q_j + \beta_h - \beta_{h'} - \mathbb{1}_{\{\sigma\}}(h)\gamma - W^{\text{tot}}\delta_{hh'} \tag{3a}$$

$$\text{s.t.} \quad \sum_{j\in V^{\text{sto}}} q_j \leq Q \tag{3b}$$

$$\overline{q}_j \sum_{(l,j)\in A} x_{lj} \geq q_j \qquad \forall j \in V^{\text{sto}} \tag{3c}$$

$$\sum_{(l',l)\in A} x_{l'l} - \sum_{(l,l')\in A} x_{ll'} = 0 \qquad \forall l \in V \setminus V^{\text{fac}\sigma} \tag{3d}$$

$$\sum_{(h,l)\in A} x_{hl} = 1 \tag{3e}$$

$$\sum_{(l,h')\in A} x_{lh'} = 1 \tag{3f}$$

$$f_{ll'} \leq Q x_{ll'} \qquad \forall (l,l') \in A \tag{3g}$$

$$\sum_{(l,l')\in A} f_{ll'} - \sum_{(l',l)\in A} f_{l'l} = \begin{cases} q_l & \forall l \in V^{\text{sto}} \\ 0 & \forall l \in V \setminus (V^{\text{sto}} \cup V^{\text{fac}\sigma}) \end{cases} \tag{3h}$$

$$\sum_{(h,l)\in A} f_{hl} = 0 \tag{3i}$$

$$\sum_{(l,h')\in A} f_{lh'} = \sum_{j\in V^{\text{sto}}} q_j \tag{3j}$$

$$q_j \geq 0 \qquad \forall j \in V^{\text{sto}} \tag{3k}$$

$$x_{ll'} \in \mathbb{Z}_{\geq 0} \qquad \forall (l,l') \in A \tag{3l}$$

$$f_{ll'} \geq 0 \qquad \forall (l,l') \in A \tag{3m}$$

To generate new routes in an efficient way, we rely on dynamic programming to solve a SPPRC. The goal is to find a shortest path from $h \in V^{\text{fac}\sigma}$ to $h' \in V^{\text{fac}}$ on the road-network graph $G = (V, A)$, such that the path respects the vehicle capacity and the path cost correspond to reduced cost (3a). Working with the original road-network graph can lead to significant savings in computing time as shown by Letchford et al. (2014), who present pricing routines for the VRP with time windows. We show that keeping the road-network structure is also beneficial to the Cm-CTP-IF, as nodes along a path are considered and can help to create better routes (Section 8.2). To reduce the number of nodes examined, we simplify G by removing nodes that do not belong to $V^{\text{sel}} \cup \{h, h'\}$ and are not intersection nodes (i.e., nodes with in degree smaller than 2), and reconnecting their predecessors and successors by assuming a shortest path distance on these arcs. In G, a path is allowed to visit any node $l \in V$ several times which is why it can contain cycles. However, the demand $q_j \ \forall j \in V^{\text{sto}}$ can only be collected once, which is why we face a special SPPRC, where elementarity constraints only apply in some cases. For each node $j \in V^{\text{sel}}$ with $\overline{q}_j > 0$, we introduce a duplicate node j' that represents the collection of the associated demand and is therefore subject to the elementarity constraints. More precisely, a node $j \in V^{\text{sel}}$ can be visited several times but its duplicate j' at most once. Furthermore, we forbid to split the demand in the PP to speed-up the resolution, so only routes collecting demand $q_j = \overline{q}_j \ \forall j \in V^{\text{sto}}$ are considered. This restriction should not make an important difference in the final solution as intermediate facilities are close to the demand nodes (.i.e, splitting the demand is less efficient) and as proven by the experimental comparisons with other approaches (see Section 8.4).

More formally, a label L represents a partial path in G starting from the source node h, where $\overline{c}(L)$ corresponds to its reduced cost, $q(L)$ represents the accumulated collected demand and $U(L) \subseteq V^{\text{sel}}$ is the set of nodes visited and satisfied along the path. Then, we apply the concept of *ng*-path which was introduced as a new route relaxation for the capacitated VRP (CVRP) and the CVRP with time windows (Baldacci et al. 2011). *Ng*-paths are a compromise between elementary and non-elementary paths and are built according to customer sets (i.e., *ng*-sets) which are associated with each customer and often contain neighbours within a short travelling distance. The larger the size of these sets, the closer the *ng*-path is to an elementary path. Following that, we define for each collection node $j' \in V^{\text{sel}}$ a neighbourhood $N_{j'} \subseteq V^{\text{sel}}$ that contains the closest nodes to j' in G and itself such that $|N_{j'}|$ is lower than a given *ng*-size. The set $U(L)$ is replaced by a subset $\Pi(L)$ which is empty at the source node h and then built recursively so that it only contains nodes in N_j of the last visited node $j \in U(L)$. A label L can be extended to a collection node $j' \in V^{\text{sel}}$ along an arc $(l, j') \in A$ to collect the demand $q_{j'}$ if $j' \notin \Pi(L)$. When extended, a new label L' is created with $\Pi(L') = (\Pi(L) \cap N_{j'}) \cup \{j'\}$. Because the cardinality of $\Pi(L')$ is less than or equal to $|N_{j'}|$, the replaced condition is less restrictive yielding generally more dominated labels and faster computation times. We believe that the concept of *ng*-path is even stronger when applied to road-network graphs, where labels can only be extended locally to the nearest intersection nodes and not to every neighbor as in the complete-graph representation, and thus, *ng*-sets of lower cardinality are required to ensure elementarity constraints (as shown in Section 8.2).

7.3. Feedback mechanism

To connect the two phases (SCP and CVRP-IF) with each other, we provide some feedback from the routing solution to the candidate locations which is then considered in the creation of the MST (Section 6.2) in the next iteration. More precisely, we use the dual variable values ξ_j of constraint (2b) provided by the RMP (Section 7.1) in the last iteration of column generation, which gives an estimated cost for collecting a unit of demand at node $j \in V^{\text{sto}}$ given the set of routes R, and associate it with each node $j \in V^{\text{sel}}$. Recall the reduced graph $G^{\text{D}} = (V^{\text{I}} \cup W, E)$, where $V^I \subset V$ is the set of road intersections and E is a set of undirected edges that reconnects the nodes in G^{D} based on the road network information in G. Let $V_e^{\text{sto}}\ \forall e \in E$ be the set of candidate locations that either lie between the two end nodes of e in G (i.e., those that have been removed to build G^{D}) or on one of its end nodes. We then define a new edge weight $\overline{w_e} = \sum_{j \in V_e^{\text{sto}}} \xi_j / |V_e^{\text{sto}}|\ \forall e \in E$ that will be provided as a feedback to the clique cover heuristic (Algorithm 2 in algorithm 2), so that in the first phase of a next iteration, the MST is defined in a way to avoid cliques of demand nodes that result in the selection of candidate locations with a high cost. This mechanism will first lead to diversification as there is no information available for candidate locations that have not been selected in any solution. Over time more candidate locations will be chosen and associated with a cost ξ_j so that the creation of set covers will tend to improve with respect to the routing cost.

7.4. Diving heuristic

To improve the best solution obtained from the solution method (Algorithm 1), we apply a diving heuristic (Joncour et al. 2010) at the end of the algorithm (Algorithm 1) in which additional columns may be generated to improve the integer routing solution. More precisely, if the optimal solution of the RMP is fractional, the heuristic applies a depth-first search until an integer solution is found. The heuristic repetitively fixes one fractional variable $x_r = 1$ corresponding to an elementary route and uses column generation to include promising additional routes. At the end of the heuristic, a MILP with an extended set of columns set as binary variables (i.e., $x_r \in \{0, 1\}$) is solved which on average leads to better integer solutions as shown in Section 8.2.

8. Computational experiments

In this section, we present the results of the computational experiments. The solution approach has been implemented in Java. To solve the MILP and the RMP we used the Gurobi 9.5.2 MIP solver via its Java API. The instances were tested on a computer with a Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz processor, 32 GB of RAM, operating under Linux, and a time limit of 12 hours was set.

8.1. Problem instances

We use the problem instances introduced by Fischer et al. (2023b) which are made available online under the below link[1]. They are defined with respect to the number of demand nodes $|W| \in \{50, 100, 200, 600\}$ and

are labelled accordingly. The three main parameters that characterize an instance are the maximum walking distance $w \in \{50, 100, 200, 300\}$, the number of routes $m \in \{5, 10, 20\}$, and the number of intermediate facilities $f \in \{1, 2, 3\}$, whose locations are given. This results into 36 instances for each dataset, which yields 144 instances in total. Finally, we assume a stop penalty value of $t^{\text{sto}} = 5$ s. Since we do not allow to split demand in the collection process, we preprocess each demand node $i \in W$ with $d_i > Q$, by distributing its demand in excess to its closest nodes $l \in V \setminus W$ and add each such l to the set of demand nodes W.

8.2. Analysis of column generation

This section presents some statistics of the proposed column generation approach for the CVRP-IF. They have been generated on a subset of the instances (i.e., benchmark instances) selected to be as representative as possible over the datasets by keeping all parameters but one fixed. More precisely, for the datasets $W50, W100$, and $W200$, we define $w = 50, m = 10$ and $f = 2$ as a starting point and then vary only one of the three parameter values at a time, resulting in 24 instances in total. For each of the instances, a set cover V^{sel} is built and used for the different tests. To evaluate the benefit of the road-network graph G in the PP where a SPPRC is solved by dynamic programming (Section 7.2), we introduce a complete directed graph $G' = (V', A')$ which is made up by the node set $V' = V^{\text{sel}} \cup \{h, h'\}$ and the arc set A' such that an arc $(j, j') \in A'$ represents a shortest path from $j \in V'$ to $j' \in V'$ of length $\ell_{jj'}$. For the *ng*-path approach, we set a minimal *ng*-size ratio $\rho = 10\%$ to compute the *ng*-size for each instance with $\lceil \rho |V^{\text{sel}}| \rceil$ (i.e., a percentage ρ of the respective set cover size).

As shown in Table 1, with the road-network graph all instances can be solved within the time limit, while with the complete graph no optimal solution can be found for the instance $W200, w = 50, m = 5, f = 2$. On average over the instances that both graph representations were able to solve, the RMP is solved faster with the road-network graph G, and less iterations and columns are needed, confirming the benefit of using the road-network graph G in the PP. Note that the average and median improvements of the computational time show consistent improvements for all 23 instances when solving with the road-network graph (the minimum improvement is 29.64%).

Based on the same benchmark instances, we validate the concept of *ng*-path in the SPPRC and decide on the size of the associated *ng*-sets for the subsequent computational experiments. Table 2 presents some statistics on various sizes of such sets, with the minimal *ng*-size ratio $/rho$. The benefit of applying the concept in general is clearly visible when comparing the number of instances solved for any value of ρ with respect to not using the concept at all (i.e., $\rho = 100\%$). On average over the instances that all ρ values were able to solve, the linear program (LP) of the RMP is solved faster with decreasing values of ρ. However, smaller values of ρ report weaker bounds on the routing cost, as shown in the gaps to the best objective values reported by the various settings of ρ, due to an increased ratio of non-elementary columns in the LP.

[1] https://drive.switch.ch/index.php/s/unpTFHxEwccSXRl

Table 1 Comparison of the performance of solving RMP when using a complete graph (G') or the road-network graph (G) (average values are used when not specified) to solve the SPPRC by dynamic programming (Section 7.2).

Graph representation	G'	G
# Instances solved	23	**24**
Comp. time (s)	4074.99	**208.41**
Improvements of comp. time (%)	-	**78.41%**
Median improvements of comp. time (%)	-	**84.16%**
# Iterations	29.04	**20.57**
# Columns	4202.87	**2902.26**

As a consequence, we believe that a *ng*-size ratio of 10% gives the best trade-off between lower bounds and computation times and decide to use it for our experiments.

Table 2 Validation of the concept of *ng*-path to solve the SPPRC by dynamic programming (Section 7.2) and performance of solving RMP with respect to various minimal *ng*-size ratios ρ.

Minimal *ng*-size ratio ρ (%)	5%	10%	15%	20%	100%
# Instances solved	**24**	**24**	23	23	17
Average *ng*-size ratio (%)	**8.54%**	11.56%	17.13%	21.89%	100.00%
Comp. time (s)	**5.41**	**5.41**	6.05	6.84	1862.88
Gap to best obj. value (%)	0.10%	0.02%	0.01%	**0.00%**	0.00%
Non-elem. cols. (%)	12.28%	9.36%	**5.77%**	7.84%	0.00%
# Iterations	**9.94**	10.59	10.29	11.06	10.18
# Columns	1508.24	**1457.41**	1473.94	1517.35	1368.65

Table 3 represents statistics of the column generation approach for the benchmark instances on datasets $W50, W100$ and $W200$. As can be seen from it, on average most of the time is spent in the PP where a SPPRC is solved by dynamic programming (Section 7.2). We observe that the RMP LP, compared to the RMP MILP, takes up on average the biggest parts with respect to the computation time, and with increasing dataset size (i.e., number of demand nodes $|W|$), more iterations and columns are needed on average to solve the instances. Furthermore, the approach reports relatively small average integrality gaps, which confirm the applicability of the reformulation to estimate the routing cost. Finally, these integrality gaps are smaller (i.e., 0.61% on average over all instances) than the ones reported before applying the diving heuristic (Section 7.4) at the end of the algorithm (i.e., 2.24% on average over all instances), and therefore shows the benefit of using this heuristic.

8.3. Analysis of the developed decomposition method

In this section, we validate the feedback and the diversification mechanisms used in the developed decomposition method. As explained in Section 7.3, we provide some feedback from the routing solution to the candidate locations which is then considered in the creation of the MST (Section 6) in the next iteration by updating the edge weights. On the same benchmark instances as introduced in Section 8.2, we tested various

Table 3 Statistics of the column generation approach (i.e., average values) for the benchmark instances on datasets $W50, W100$ and $W200$.

Column generation	*W50*	*W100*	*W200*
Comp. time RMP (%)	2.23%	0.71%	0.37%
Comp. time PP (%)	97.77%	99.29%	99.63%
Comp. time RMP LP (%)	99.76%	99.90%	98.45%
Comp. time RMP MILP (%)	0.24%	0.10%	1.55%
Int. gap RMP (%)	0.08%	0.48%	1.27%
Int. gap RMP before Diving (%)	1.53%	2.08%	3.12%
# Iterations	14	15	93
# Columns	1768	2646	7491

fixed values of α to see the impact of the feedback mechanism in general and decide on the value of α for the subsequent experiments. Figure 4 visualizes for each value of α (i.e., label next to the point) the average gap of the objective value to the best solution (%) and the average computation time ($\cdot 10^3$ s) when the best solution is found. This plot clearly shows the benefit of applying the feedback mechanism as $\alpha = 0$ (i.e., not using the feedback mechanism) reports the highest average gap to the best solution (i.e., 0.9%) while $\alpha = 1$ (i.e. using the feedback mechanism considering only the previous routing solution) has the lowest average gap to the best solution (i.e., 0.5%) with approximately the same average computational time (i.e., 8400 s). As the other α values that are a bit faster in finding the best solution (i.e., 0.9%, 0.2% and 0.6%) are worse with respect to the average gap to the best solution (i.e., 0.77%, 0.7%, 0.78%, respectively), we decide to use the initial value of $\alpha = 1$ for our experiments which also reports the best solution for most instances (i.e., 7/24, other α values # is best ≤ 4).

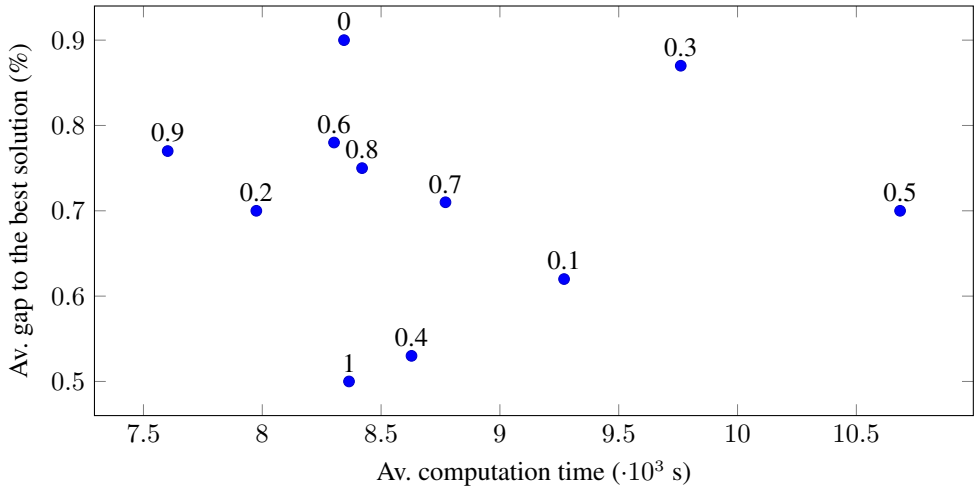


Figure 4 Analysis of the value of α (i.e., label next to the point) in the feedback mechanism (Section 7.3) with respect to the average computation time ($\cdot 10^3$ s) and average gap to the best solution (%).

The two diversification mechanisms used in the decomposition method (Section 5) are 1) to start at a random node in the intersection graph G^{I} when finding a minimum clique cover and 2) to change the value of α in the feedback mechanism. Note that the second mechanism is only applied when the first does not produce a new set cover. On the same benchmark instances as above (Section 8.2), Table 4 presents the number of instances for which each of the diversification mechanisms was used together with the average integrality gaps (%) before and after applying the mechanisms. With increasing size of the instances (i.e., number of demand nodes $|W|$), the diversification mechanisms are less used as there exist more set covers in total which can easily be found with the default setting of the clique cover approach (Section 6). The first mechanism (i.e., clique cover) is used more often and leads to an average decrease of 1.27% of the integrality gap, while the second mechanism (i.e., alpha value) reports an average decrease of 0.25% of the integrality gap for the smallest dataset. Since both mechanisms on average lead to an improvement when applied (even if the alpha value diversification does not seem to add much value), we decide to keep them in the decomposition method for the subsequent experiments as both do not require additional computations when not used.

Table 4 Analysis of the diversification mechanisms used in the decomposition method (Section 6).

Diversification mechanisms	***W50***	***W100***	***W200***
# instances	8	8	8
Av. int. gap (%)	0.08%	0.48%	1.27%
# clique cover diversification (ccd) used	8	8	3
Av. int. gap ccd used (%)	0.08%	0.48%	0.08%
Av. int. gap before ccd (%)	1.63%	1.66%	0.87%
# alpha value diversification (avd) used	3	0	0
Av. int. gap avd used (%)	0.14%	-	-
Av. int. gap before avd (%)	0.39%	-	-

8.4. Comparison of the solution methods.

In this section, we assess the performance of the developed solution method and first compare it with an additional approach that uses the set construction method *constructSet()* presented in Fischer et al. (2023b) for the SCP. Given a set cover V^{sel}, we then apply column generation (Section 7) to solve the CVRP-IF as in our solution method. Algorithms 2 - 3 are replaced by the procedure *constructSet()* that generates a set cover by iteratively adding nodes until the resulting set is a set cover. This procedure first selects a demand node at random and then chooses a candidate location that covers this demand node and maximizes the total number of demand nodes that are covered. Finally, it checks for redundant nodes, that is, nodes that can be removed from V^{sel} while it continues to be a set cover. We refer to this approach as the *Iterative* method. In Table 5, we observe that for some of the instances of the largest dataset (i.e., $W600$) both methods were not able to find a solution within the time limit. This is mainly because of the difficulty of the routing problem which

used more than 90% of the time on average. On average over the instances that both methods were able to solve, the *Clique* method clearly outperforms the other method with respect to the number of best solutions and the average gaps of the objective value to the best solution found by the two methods. Furthermore, the *Clique* method was able to find and treat more set covers on average. For the datasets $W50$ and $W200$ the *Iterative* method returned its best solution faster than the *Clique* method, and for the largest dataset (i.e., $W600$) the *Iterative* method needed less iterations to find its best solution. Nevertheless, those differences in time and number of iterations are in general small compared to the superior solution quality of the *Clique* approach.

Table 5 Comparison of the solution methods based on results aggregated over the datasets $W50, W100, W200,$ and $W600$.

	W50		***W100***		***W200***		***W600***	
	Iterative[1]	**Clique**	**Iterative**[1]	**Clique**	**Iterative**[1]	**Clique**	**Iterative**[1]	**Clique**
# feasible solutions	36	36	36	36	36	36	21	21
# is best	6	**35**	8	**29**	9	**27**	8	**13**
Av. gap to best solution (%)	0.75%	**0.01%**	0.49%	**0.06%**	0.45%	**0.29%**	0.48%	**0.29%**
Av. # treated set covers	25349	**30922**	36943	**39296**	18008	**18127**	1022	**1035**
Av. time best solution (s)	**5246**	10617	17315	**13383**	**14718**	15055	17532	**16796**
Av. iteration best solution	69780	**22969**	72050	**21774**	8346	**8013**	**354**	389
Av. time SCP (%)	1.82%	31.24%	1.94%	5.89%	1.44%	5.97%	0.64%	3.29%
Av. time CVRP-IF (%)	98.18%	68.76%	98.06%	94.11%	98.56%	94.03%	99.36%	96.71%

Fischer et al. (2023a) propose a heuristic Benders decomposition approach to solve the Cm-CTP-IF in which a SCP is solved in the master problem and a CVRP-IF is solved with column generation in the subproblem. In this heuristic, the Benders cuts that are added to the master problem are approximated, as deriving valid Benders cuts leads to a dramatic increase in computation time. We refer the reader to Fischer et al. (2023a) for more details on their approach. For the instances used in their and our experiments, namely, $|W| \in \{50, 100, 200\}, w \in \{100, 200, 300\}, m \in \{5, 10, 20\}, f \in \{1, 2, 3\}$, Table 6 shows the number of feasible solutions, and for the instances that all methods were able to solve, the number of times each method returned the best solution and the average gap to the best solution (%). Theses results clearly show the superiority of the *Clique* approach compared to all previous methods for the Cm-CTP-IF as it returned the best solution for most instances (i.e., 48/51) and reported the lowest gap to the best solution on average (i.e., 0.01%).

9. Conclusion

In this article, we formulated the Cm-CTP-IF which is inspired by a waste collection problem with intermediate disposal facilities to accommodate the usage of smaller vehicle types. It is a variation of the Cm-CTP-R introduced in Fischer et al. (2023b) which is closely related to the m-CTP. We extended their proposed

[1] Fischer et al. (2023b)

Table 6 Comparison of the solution methods with respect to the MILP formulation (Section 4) and the heuristic Benders decomposition approach (Fischer et al. 2023a) on a subset of instances (i.e., $|W| \in \{50, 100, 200\}, w \in \{100, 200, 300\}, m \in \{5, 10, 20\}, f \in \{1, 2, 3\}$).

	MILP	Benders	Iterative	Clique
# feasible solutions	51	**81**	**81**	**81**
# is best	0	0	7	**48**
Av. gap to best solution (%)	10.59%	6.37%	0.70%	**0.01%**

road-network-based MILP formulation and developed a solution method that decomposes the problem into a SCP and a CVRP-IF to handle larger instances derived from real-life data. We applied column generation to solve the CVRP-IF and presented a novel approach for the SCP that exploits the structure of the problem to efficiently find a set cover by using a graph theoretical approach. We also compared our method with an approach that uses the set construction method presented in Fischer et al. (2023b) and a heuristic Benders decomposition approach introduced in Fischer et al. (2023a).

The results show that the Cm-CTP-IF is a difficult problem and for some of the largest instances no solution could be found within the time limit. Nevertheless, our solution method which includes validated feedback and diversification mechanisms clearly outperforms the other existing methods with respect to the number of best solutions and the average gaps to the best solution. The proposed road-network structure and the concept of *ng*-path are beneficial to solve the routing problem faster, which has an impact on the computational efficiency as most of the time is consumed in the CVRP-IF or more precisely in the PP.

Further research will be needed to improve the dynamic program for solving the PP so that less time is spent on that problem and more set covers can be evaluated within the time limit. For instance, to speed up the dynamic program, the road network could be exploited by considering promising subsets of the selected candidate locations that lie between the respective pair of facilities (i.e., start and end facilities) of each PP, resulting in routes that are well suited for each pair. On the graph theoretical aspect, it could be interesting to study whether the intersection graph of subtrees of a planar graph (instead of a tree) has some structure that could be exploited. Indeed, it is reasonable to assume the walkable network to be planar. The goal would therefore be to directly apply some minimum clique cover algorithm to the whole graph, without the need to restrict to a selected spanning tree.

Acknowledgments

The authors gratefully acknowledge the support of Innosuisse under grant 36157.1 IP-EE. They also thank the Hasler Foundation for the support of the research stay in Montreal, Canada and they highly value the involvement of Schwendimann AG in conducting practical experiments and providing the associated data.

References

Allahyari S, Salari M, Vigo D (2015) A hybrid metaheuristic algorithm for the multi-depot covering tour vehicle routing problem. *European Journal of Operational Research* 242(3):756–768.

Angelelli E, Speranza MG (2002) The periodic vehicle routing problem with intermediate facilities. *European Journal of Operational Research* 137(2):233–247.

Baldacci R, Mingozzi A, Roberti R (2011) New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research* 59(5):1269–1283.

Ben Ticha H, Absi N, Feillet D, Quilliot A (2018) Vehicle routing problems with road-network information: State of the art. *Networks* 72(3):393–406.

Clarke G, Wright JW (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research* 12(4):568–581.

Crevier B, Cordeau JF, Laporte G (2007) The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research* 176(2):756–773.

Dantzig GB, Wolfe P (1960) Decomposition principle for linear programs. *Operations research* 8(1):101–111.

Fischer V, Legrain A, Schindl D (2023a) A benders decomposition approach for a capacitated multi-vehicle covering tour problem with intermediate facilities, university of Fribourg (CH), Polytechnique Montreal (CA), Haute Ecole de Gestion (CH).

Fischer V, Paneque MP, Legrain A, Bürgy R (2023b) A capacitated multi-vehicle covering tour problem on a road network and its application to waste collection. *European Journal of Operational Research* ISSN 0377-2217.

Gavril F (1974) The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B* 16(1):47–56.

Golumbic MC (2004) *Algorithmic graph theory and perfect graphs* (Elsevier).

Hachicha M, Hodgson MJ, Laporte G, Semet F (2000) Heuristics for the multi-vehicle covering tour problem. *Computers & Operations Research* 27(1):29–42.

Helly E (1923) Über mengen konvexer körper mit gemeinschaftlichen punkte. *Jahresber. Dtsch. Math.-Ver* 32:175–176.

Joncour C, Michel S, Sadykov R, Sverdlov D, Vanderbeck F (2010) Column generation based primal heuristics. *Electronic Notes in Discrete Mathematics* 36:695–702.

Kruskal JB (1956) On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society* 7(1):48–50.

Letchford AN, Nasiri SD, Oukil A (2014) Pricing routines for vehicle routing with time windows on road networks. *Computers & Operations Research* 51:331–337.

Letchford AN, Nasiri SD, Theis DO (2013) Compact formulations of the Steiner traveling salesman problem and related problems. *European Journal of Operational Research* 228(1):83–92.

Markov I, Varone S, Bierlaire M (2016) Integrating a heterogeneous fixed fleet and a flexible assignment of destination depots in the waste collection VRP with intermediate facilities. *Transportation Research Part B: Methodological* 84:256–273.

Muter I, Cordeau JF, Laporte G (2014) A branch-and-price algorithm for the multidepot vehicle routing problem with interdepot routes. *Transportation Science* 48(3):425–441.

Nedjati A, Izbirak G, Arkat J (2017) Bi-objective covering tour location routing problem with replenishment at intermediate depots: Formulation and meta-heuristics. *Computers & Industrial Engineering* 110:191–206.

Ramos TRP, Gomes MI, Barbosa-Póvoa AP (2020) A new matheuristic approach for the multi-depot vehicle routing problem with inter-depot routes. *OR Spectrum* 42(1):75–110.

Rose DJ, Tarjan RE (1975) Algorithmic aspects of vertex elimination. *Proceedings of the seventh annual ACM Symposium on Theory of Computing*, 245–254.

Tarantilis CD, Zachariadis EE, Kiranoudis CT (2008) A hybrid guided local search for the vehicle-routing problem with intermediate replenishment facilities. *INFORMS Journal on Computing* 20(1):154–168.