European Journal of Operational Research 315 (2024) 338–353

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Innovative Applications of O.R.

A capacitated multi-vehicle covering tour problem on a road network and its application to waste collection

Vera Fischer [a,*], Meritxell Pacheco Paneque [a], Antoine Legrain [b,c,d], Reinhard Bürgy [e]

[a] University of Fribourg, Boulevard de Pérolles 90, 1700 Fribourg, Switzerland
[b] Polytechnique Montréal, P.O. Box 6079, Station Centre-Ville, Montréal (Québec), Canada
[c] CIRRELT - Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, 2920 Chemin de la Tour, Montréal (Québec), Canada
[d] GERAD - Group for Research in Decision Analysis, 2920 Chemin de la Tour, Montréal (Québec), Canada
[e] Lucerne University of Applied Sciences and Arts, Suurstoffi 1, 6343 Rotkreuz, Switzerland

ARTICLE INFO

Keywords:
Combinatorial optimization
Waste collection
Multi-vehicle covering tour problem
Mixed-integer linear programming
Road network

ABSTRACT

In most Swiss municipalities, a curbside system consisting of heavy trucks stopping at almost each household is used for non-recoverable waste collection. Due to the many stops of the trucks, this strategy causes high fuel consumption, emissions and noise. These effects can be alleviated by reducing the number of stops performed by the collection vehicles. One possibility consists of selecting a subset of candidate locations that are scattered throughout the municipality to place collection points which are used by residents to bring their waste. Provided that the underlying road network is available and that the collection vehicle has a known capacity, we refer to this problem as the capacitated multi-vehicle covering tour problem on a road network (C*m*-CTP-R). We propose a road-network-based mixed-integer linear programming (MILP) formulation that exploits the sparsity of the network. We compare it against the MILP formulation that results from assuming a customer-based graph, which is typically used in vehicle routing problems (VRP). To solve large instances, we develop a two-phased heuristic approach that addresses the two subproblems the C*m*-CTP-R is built on: a set covering problem to select the locations and a split-delivery VRP to determine the tours. Computational experiments on instances derived from real-life data show that the road-network-based formulation is better suited. Furthermore, the proposed heuristic provides good solutions with optimality gaps below 1.7% and finds better solutions for most of the instances that the exact method is not able to solve within a given time limit.

1. Introduction

Waste collection is an important process in waste management. It mainly involves the transportation of waste from collection sites to disposal facilities, and represents one of the primary and most expensive logistical activities performed by any municipality. Indeed, collection costs of municipal solid waste often account for up to 70% of the total waste management budget (Tavares et al., 2009). The design and operation of a waste collection system is a difficult task, since it entails multiple distinguishing features, such as the types of collected waste (e.g., dry recyclable, wet food or residual), the collection frequency (e.g., weekly, bi-weekly or on-demand), and the pricing (e.g., weight- or volume-based). Other key aspects are the containers or bags being used at collection sites and the collection vehicles. Furthermore, residents have high expectations when it comes to waste collection, in the sense that they aim at a frequent collection at a site close to their home but are not willing to spend neither money nor time for it.

From the residents' point of view, collection methods are often divided into curbside (pick-up) systems, where the waste is disposed outside their property, and bring (drop-off) systems, where the waste is brought to communal collection sites (Rodrigues et al., 2016). Curbside systems are used in most Swiss municipalities to collect non-recoverable waste, with heavy trucks stopping at almost each household. They are the most convenient for residents. However, due to the nature of the employed trucks and the number of performed stops, this strategy results into large collection times and causes negative effects such as high fuel consumption, emissions and noise. In hopes of designing a more efficient and sustainable residential waste collection system, we investigate in this paper the location of collection points that are close to residential buildings and can be easily accessed by foot. These collection points represent physical areas (e.g., a circle painted on the ground) where residents must leave their bags.

* Corresponding author.
E-mail addresses: vera.fischer@unifr.ch (V. Fischer), meritxell.pacheco@unifr.ch (M. Pacheco Paneque), antoine.legrain@polymtl.ca (A. Legrain), reinhard.buergy@hslu.ch (R. Bürgy).

https://doi.org/10.1016/j.ejor.2023.11.040
Received 20 December 2021; Accepted 25 November 2023
Available online 29 November 2023

This problem can be formulated as a facility location problem (FLP; e.g., Ghiani et al., 2012; Tralhão et al., 2010). Nevertheless, since the location of collection points is interlaced with the subsequent collection tours, both decisions should be simultaneously tackled by means of location-routing problems (LRPs; Prodhon & Prins, 2014). These problems represent an approach to model and solve locational problems while paying special attention to the underlying issues of vehicle routing (Nagy & Salhi, 2007). Despite the different nature of the decisions being addressed (location is strategical whereas routing is tactical/operational), the overall system cost may be excessive if they are handled separately. Additionally, the use of LRPs could decrease the total costs over a long planning horizon within which routes might change (Salhi & Nagy, 1999).

Given the location of a single disposal facility, a set of candidate locations, a single collection vehicle, and the amount of waste produced at each residential building in the time horizon under consideration, we aim at selecting a subset of candidate locations to place collection points while determining the tours that visit them in order to collect the waste and transport it to the disposal facility. We assume that the waste gathered in a collection point can be split and transported in different tours. The number of tours is determined by the total amount of waste to be collected and the capacity of the collection vehicle. For each residential building, a given rank sorts the candidate locations in compliance with some criterion (e.g., in increasing order of walking distance). The goal of the problem is to locate collection points such that all residential buildings are covered with minimum total travel time. We say that a residential building is covered if its waste is gathered at a collection point from its rank. Besides, we assume that residential buildings will consider the highest-ranked candidate location where a collection point is placed.

As defined, this problem is closely related to the multi-vehicle covering tour problem (m-CTP; Hachicha et al., 2000). The m-CTP can be seen as an LRP that generalizes the vehicle routing problem (VRP). We formulate our problem as a variant of the m-CTP in which the constraints on the length and number of vertices of each tour are replaced by vehicle capacity constraints. Furthermore, each residential building does not only have to be covered by a collection point from its rank, but also must be allocated to the highest-ranked collection point that belongs to the solution. Since we assume that the underlying road network is known, we refer to this problem as the capacitated m-CTP on a road network (Cm-CTP-R).

In this paper, we propose a compact mixed-integer linear programming (MILP) formulation for the Cm-CTP-R. This formulation relies on a road-network graph, a more detailed approach to represent the road network that is also computationally more efficient than the so-called customer-based graph typically used in VRPs (Letchford et al., 2014). In a customer-based graph, a node is introduced for each customer and each depot, and an arc represents the shortest path between the start node and the end node. Instead, the proposed formulation exploits the sparsity of the road network by introducing decision variables for each road segment. This technique has already been considered for other problems, such as the Steiner traveling salesman problem (STSP) in Letchford et al. (2013), and it is extended here for the Cm-CTP-R. We also formulate the problem that results from assuming a customer-based graph on the network for a numerical and computational comparison of the two approaches.

To handle the large instances of the Cm-CTP-R that general-purpose MILP solvers might fail to solve, we develop a two-phased heuristic method. It is based upon solution procedures for the two interdependent subproblems the Cm-CTP-R is built on: a set covering problem (SCP) to identify the subset of candidate locations and the split-delivery VRP (SDVRP) to generate the tours that visit such locations. In the first phase, we construct set covers and approximate the routing cost associated with each of them by means of a giant tour that is gradually constructed by visiting either a candidate location or one of its alternatives. An alternative to a candidate location represents another candidate location that can cover the same residential buildings. In the second phase, we solve a SDVRP on the candidate locations visited in the giant tour. To do so, we transform the problem into a capacitated VRP (CVRP) with an a priori splitting strategy of the nodes with a large amount of waste (Chen et al., 2017), and we solve the resulting problem with the state-of-the-art metaheuristic for CVRP proposed by Vidal (2022). To conduct the experiments, we derive a set of instances for the Cm-CTP-R by relying on data from various Swiss municipalities. In particular, we compare the road-network-based and customer-based MILP formulations, validate and assess the performance of the heuristic method and analyze the travel time savings of the introduced waste collection system with respect to the state of practice.

To sum up, the contribution of this research is threefold:

1. We introduce a variant of the m-CTP, the Cm-CTP-R, in which the constraints on the length and number of nodes of each tour are replaced by vehicle capacity constraints, and the coverage of nodes is ensured by the allocation of residential buildings to a collection point that appears in their rank. We assume a rank that sorts the candidate locations according to a given criterion and define operational constraints that allocate residential buildings to their highest-ranked candidate location where a collection point is placed. This problem is a real-life problem motivated by the collaboration with an industrial partner. To test the proposed methodology, we derive a set of instances based on real-life data that are appropriate to the Cm-CTP-R with more than 2000 nodes that can be visited and up to 600 nodes to cover. These instances are made available online to other researchers to encourage future research on the introduced problem.[1]
2. We show that the road-network-based MILP formulation of the Cm-CTP-R, which relies on a more detailed representation of the underlying street network, is both computationally more efficient and superior with respect to solution quality than the MILP formulation that relies on the associated customer-based graph. This is not an obvious claim as it is typically not the case in the literature.
3. To handle large instances, we propose a two-phased heuristic method that exploits the decomposable structure the Cm-CTP-R is built on. This method is tested against the road-network-based MILP formulation for the instances solved to optimality, providing the optimal solution for most of them, as well as for the instances not solved to optimality, providing a better solution than the exact method in a given time budget for most of them.

The remainder of the paper is organized as follows. Section 2 provides an overview of relevant works in the context of m-CTP and road-network representation. Section 3 formally defines the problem and the particularities for its application in waste collection, and enumerates the modeling assumptions we make. Sections 4 and 5 present the MILP formulations and the two-phased heuristic method, respectively. Section 6 reports the computational and numerical experiments, and Section 7 summarizes the main findings and discusses avenues for future research.

2. Related work

Despite the increasing attention received by LRPs in the last years (Schneider & Drexl, 2017), relatively few papers on the covering tour problem (CTP) and derivatives thereof have been published. The first appearance of the CTP can be attributed to Current (1981). The CTP is formally defined in Gendreau et al. (1997) as the problem of finding a Hamiltonian tour with minimum length over the subset of nodes to

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

be visited such that each node in the subset of nodes to be covered lies within a prespecified distance from a tour node. A two-index formulation is developed and solved exactly with a branch-and-cut algorithm. Similar to Current and Schilling (1989), they further propose a heuristic approach that combines a SCP and a Traveling Salesman Problem (TSP) heuristics. Baldacci et al. (2005) formulate the CTP as a two-commodity network flow problem and propose three scatter-search heuristic algorithms.

The extension of the CTP to multiple vehicles, the m-CTP, is defined in Hachicha et al. (2000) as the problem of designing up to m vehicle tours starting and ending at the depot with minimum total length such that the nodes to cover lie within a preset distance of a tour node and both the number of nodes and the length of any tour do not exceed given values. The authors point out that the m-CTP appears to be more difficult than the VRP and therefore heuristics might be the only viable methods to find good solutions for practically relevant instances. They propose three heuristics called modified savings, modified sweep and route-first/cluster-second that are partially based on the corresponding methods for the standard VRP. All of them allow to find good solutions for realistic instances within a reasonable computational time, being the modified savings the fastest one and the other two better in terms of solution quality.

Jozefowiez (2014) introduces a branch-and-price algorithm based on a column generation approach to solve the m-CTP exactly. The master problem is a SCP and the subproblem is a variant of the profitable tour problem (PTP; Dell'Amico et al., 1995) solved by a branch-and-cut algorithm. This methodology is tested on instances with up to 60 nodes that can be visited and up to 150 nodes to cover. Ha et al. (2013) propose a two-commodity flow formulation based on the formulation of Baldacci et al. (2005) for the m-CTP when the length constraint is relaxed and m is a decision variable. They consider a standard branch-and-cut algorithm to exactly solve the problem. Computational results show that it outperforms the algorithm by Jozefowiez (2014) in the same context. They also develop a two-phased metaheuristic based on an evolutionary local search (ELS). In the first phase, subsets of nodes that cover all customers are created, and in the second phase, a VRP with unit demands on each subset is solved. The generated solution is within 1.5% of optimality for the considered test instances with up to 200 nodes. Kammoun et al. (2017) apply a variable neighborhood search (VNS) heuristic based on the variable neighborhood descent (VND) method for the variant without the length constraint. This method is compared against the one of Ha et al. (2013) on the same instances and reports better or equal results in a smaller computational time.

More recently, Glize et al. (2020) propose an exact method based on column generation techniques to solve the m-CTP and its bi-objective version that in addition to minimizing the total distance (length) of the tours (first objective) also aims at minimizing the maximum coverage distance (second objective). The method is compared with state-of-the-art approaches on instances from the literature (with up to 200 nodes) and for the first time, seven open instances are closed to optimality, and for six open instances the best lower bounds are improved. Other variants of the m-CTP include multiple depots with capacitated vehicles (Allahyari et al., 2015), the multi-covering of nodes (Pham et al., 2017), probabilistic coverage by the nodes to be visited (Karaoğlan et al., 2018), and speed optimization on the traversed arcs (Margolis et al., 2022).

The m-CTP finds applications in problems that concern the design of bilevel transportation networks. In these problems, only a subset of the nodes is actually visited by vehicles. Some examples include the works of Hachicha et al. (2000) on the location of mobile health-care teams in rural areas; (Labbé & Laporte, 1986) on the location of boxes for overnight mail service; and Oliveira et al. (2015) on the planning of routes for urban patrolling. Additional examples can be found in humanitarian logistics. Naji-Azimi et al. (2012) address the location of satellite distribution centers for supplying humanitarian aid throughout a disaster area. They show that only very small instances can be solved efficiently using the mathematical model. The proposed multi-start heuristic produces high-quality solutions for realistic instances in reasonable computational times. Davoodi and Goli (2019) deal with a similar problem by developing a hybrid approach that combines an exact solution method (Benders decomposition) and a fast metaheuristic (VNS) to enhance the efficiency of the overall method, as confirmed by the experiments performed on a real-life case study. Another interesting application is school bus routing. In Schittekat et al. (2013), the joint problem of bus stop selection and bus route generation is formulated in its most basic form. They characterize a MILP formulation and a parameter-free matheuristic that combines a greedy randomized adaptive search procedure (GRASP) and a VND method. Experiments on randomly-generated instances with up to 80 stops and 800 students show that the matheuristic finds most known optimal solutions much faster than the exact method.

In the context of waste collection, Cubillos and Wøhlk (2020) rely on a bi-objective LRP known as maximal CTP (MCTP) for the location of recycling drop-off stations. As introduced by Current and Schilling (1994), in a MCTP a tour must visit p nodes out of n candidate locations with the goals of minimizing the total length of the tour and maximizing the covered demand. The MCTP is a variant of the CTP in which the covering objective is replaced with a constraint that requires complete coverage. In this application, the collection costs associated with the routing are approximated with a TSP that is heuristically solved, which means that the capacity on the collection vehicles is disregarded. To handle real-life sized problems, they propose a heuristic method inspired by VNS because its use for location problems has yielded good results in the literature.

Several of the above-mentioned works rely on a customer-based graph to represent the underlying road network. These are complete graphs where a node represents a point of interest (e.g., customers, depots) and an arc represents the best path (e.g., shortest, fastest) between two nodes (Huang et al., 2017). When multiple attributes are defined on road segments (e.g., travel cost, distance), this representation can have negative consequences on the solution quality and/or efficiency (Ben Ticha et al., 2018). To address this issue, a growing number of papers investigate road-network graphs. These graphs mimic the road network by defining the arcs as the road segments and the nodes as the extremities of these segments. Letchford et al. (2014) show that significant computing time savings can be achieved with respect to customer-based graphs.

There is a collection of papers relying on road-network graphs in the context of STSP. Independently introduced by , Cornuéjols et al. (1985), Fleischmann (1985) and Orloff (1974), the aim of this problem is to find a minimum-cost cycle that visits a set of required nodes at least once. Letchford et al. (2013) propose compact formulations for the STSP on a road-network graph with a linear number of variables and constraints where decision variables are introduced for each road segment. The authors show that instances with up to 500 nodes can be solved with the developed exact branch-and-cut algorithms. Note that the STSP has been transformed into the classical TSP in Álvarez-Miranda and Sinnl (2019). Clearly, the TSP is a well studied problem and can be solved quickly with the state-of-the-art solver CONCORDE (Applegate et al., 2003), which enables to solve all instances from the literature to optimality within 20 s (most of them within a second). Road-network graphs are as well receiving increasing attention in time-dependent VRP (e.g., Ben Ticha et al., 2019).

In conclusion, the review of the literature shows that a variety of real-life problems can be modeled as m-CTPs. The Cm-CTP-R differs from the standard m-CTP in the vehicle capacity constraints, the allocation of nodes that must be covered (which are allocated to specific visited nodes) and the potential split of the demand of visited nodes among tours. Some of the reviewed applications have already addressed some of these features. Davoodi and Goli (2019) and Schittekat et al. (2013) include vehicle capacity constraints and allocation but do not

allow for splits. The allocation typically results from a criterion set by the decision maker (e.g., proximity), i.e., the decision maker imposes the location to be used, whereas we assume that residents will bring their waste to the collection point that is considered higher in the assumed rank, which can have an impact on their location. Naji-Azimi et al. (2012) consider, among others, vehicle capacity constraints and split delivery. Furthermore, motivated by the findings of Letchford et al. (2014), we rely on a road-network graph to represent the underlying street network. To the best of our knowledge, a road-network-based MILP formulation has not yet been proposed for the m-CTP or a variant thereof. To handle practically relevant instances, we develop a heuristic method that tackles the two subproblems the Cm-CTP-R is built on (similar to Gendreau et al., 1997; Jozefowiez, 2014).

3. Problem definition

We provide a formal definition of the Cm-CTP-R in Section 3.1. We discuss the specifications for its application to waste collection in Section 3.2 and enumerate our modeling assumptions in Section 3.3.

3.1. Notation, data, and problem statement

Let $G = (V \cup W, A)$ be a directed graph with two node sets V and W, and arc set A representing the (directed) road network. V includes nodes that represent candidate collection points and road intersections and W is a set of nodes with positive demand (e.g., residents who produce waste in the context of waste collection). We assume that G is strongly connected, i.e., there exists a path from each node $h \in V$ to each node $h' \in V$. For each demand node $i \in W$, its demand d_i must be satisfied at one and only one node from its rank $V_i^{\text{rank}} \subseteq V$. Note that node $i \in W$ may or may not be in V_i^{rank}. We assume that V_i^{rank} is totally ordered. The ordering reflects the assumed criterion to sort candidate locations. We assume that d_i must be satisfied at the first node in V_i^{rank} at which a vehicle stops. We denote by $\text{rank}(i, j)$ the index of node j in V_i^{rank}. Then, 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. We define $V^{\text{sto}} = \cup_{i \in W} V_i^{\text{rank}}$ as the subset of potential stopping nodes. We denote by $\sigma \in V$ the depot from which the tours depart and arrive. The arc set A represents directed road segments. Let $c_{hh'}$ be the non-negative length associated with arc $(h, h') \in A$. We assume that these lengths satisfy the triangle inequality.

For the customer-based graph representation, let $G' = (V', A')$ be the complete directed graph made up by the node set $V' = \{\sigma\} \cup V^{\text{sto}}$ 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 $\ell_{jj'}$. Note that $\ell_{jj'}$ is well-defined for each ordered pair of nodes $(j, j') \in V' \times V'$ because G is a strongly connected graph.

We assume that the demand needs to be satisfied by m vehicle tours. In particular, we assume a representative vehicle of capacity Q that drives as many tours as needed within the considered time horizon to satisfy all the demand, i.e., $mQ \geq d^{\text{tot}} = \sum_{i \in W} d_i$. We allow for splits, i.e., the total demand that needs to be satisfied at a stopping node may be split up between multiple tours. We assume that the demands are arbitrarily divisible.

A solution of the Cm-CTP-R is specified by exactly m tours on the selected stopping nodes. For each tour, we usually only record the nodes at which the vehicle stops to satisfy the demand by assuming that a vehicle travels on a shortest path from one stopping node to the next. We say that a solution covers $i \in W$ if in at least one tour the vehicle stops at some node in V_i^{rank}. A solution is feasible if it covers all demand nodes, the demand of any node $i \in W$ is satisfied at the first node in V_i^{rank} at which a vehicle stops and the vehicle capacity is not exceeded in any tour. The objective of the Cm-CTP-R is to find a feasible solution with minimum total cost. The total cost is calculated as the sum of the total cost associated with the traveled distances in the tours plus the total number of stops performed by a vehicle times t^{sto}, where t^{sto} is a given stop penalty value.

3.2. Waste collection application

The road network of the geographical area under consideration can be extracted from a mapping service (OpenStreetMap, 2022 in our case). For every road intersection, we add a node to V. The set of candidate locations V^{sto} must be specified by the decision maker (e.g., municipality). For the experiments performed in Section 6, we place candidate locations on road segments such that the distance between two locations is at most 50 m. Then, any road segment longer than 50 m is split into equal-length stretches whose lengths are less or equal than 50 m. The resulting splitting points are added to V^{sto}.

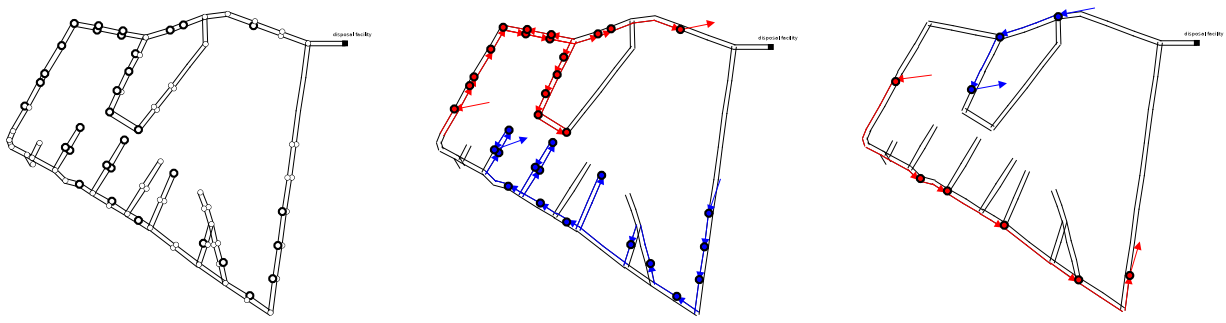
Each residential building is mapped to the node in V whose location is the closest to the building's location. This node becomes a demand node, and therefore belongs to W. Notice that multiple buildings might be represented by a single demand node. In Section 6, the demand d_i of node $i \in W$ is obtained by aggregating the average waste production per inhabitant for one week across the number of inhabitants represented by the node. We denote by γ the maximum walking distance set by government regulations for residents to bring their waste. We assume that the rank associated with residential building i (V_i^{rank}) is defined by sorting in increasing order of walking distance the candidate locations within γ. Note that the proposed modeling framework can also accommodate different ranks for different individual residents (or groups of residents) living in the same residential building. This can be achieved by simply duplicating the respective demand node as many times as individual residents are considered and defining a rank for each of them. After the ranks of all demand nodes have been determined, the candidate locations that do not belong to any rank are deleted from V^{sto}.

The depot σ corresponds to the waste disposal facility where the vehicle departs and dumps the collected waste. Whenever needed (e.g., the vehicle capacity has been reached), the vehicle can go to the disposal facility, which is connected to each candidate location. Note that visiting the disposal facility requires a considerable driving and dumping time. It should therefore only be visited when necessary. In our setting, the number of visits is equal to the number of tours.

For the sake of illustration, Fig. 1(a) presents the graph of a small neighborhood of a Swiss municipality with 57 residential buildings, 411 inhabitants and an area of 0.13 km^2. Graph G contains in total 97 nodes, out of which 33 are demand nodes, and 307 arcs, out of which 172 are incident to the disposal facility. For readability purposes, we only show the underlying undirected graph of G with left and right road sides explicitly represented. Vehicles are only allowed to turn at the so-called intersections, which are depicted in the graph as nodes in the middle of a road segment (neither on the left nor on the right road side). Furthermore, the waste disposal facility (black square), the demand nodes (bold circles) and the candidate locations (all circles) are depicted. For the disposal facility, only the closest road segments that link it to the municipality road network are drawn. Figs. 1(b) and 1(c) present two solutions for $m = 2$ tours to collect the total waste. The solutions have been obtained with the MILP formulation that relies on the road-network graph (see Section 4.1). Two values for the maximum walking distance (in meters) are considered: $\gamma = 0$ (Fig. 1(b)) and $\gamma = 100$ (Fig. 1(c)). Notice that $\gamma = 0$ corresponds to door-to-door collection. In Section 6.1 we detail the assumptions on the remaining input data. For $\gamma = 0$, in tours 1 (red) and 2 (blue) the vehicle stops at 17 and 16 collection points, respectively. The total travel and collection time is 4576 s. For $\gamma = 100$, in tours 1 and 2 the vehicle only stops at 6 and 3 collection points, respectively. The total travel and collection time is 3795 s. This represents almost a 20% decrease in the total time with respect to door-to-door collection, which results into a large gain from a practical perspective.

3.3. Modeling assumptions

In this section, we discuss the modeling assumptions we make with respect to various aspects of the problem.



(a) Example (b) $\gamma = 0$ (c) $\gamma = 100$

Fig. 1. Visualization of the illustrative example and tours 1 (red) and 2 (blue) of an optimal solution with maximum walking distance $\gamma = 0$ and $\gamma = 100$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

A1: Deterministic demand. The amount of waste associated with each demand node is known for a given time horizon, and we assume it does not change within the time horizon.

A2: Allocation of residential buildings. Given the locations of collection points, it makes sense to assume that residents will bring their waste to their highest-ranked location. Our model is capable of handling any rank of the candidate locations in compliance with a given criterion, such as walking distance or proximity to interesting points. In our experiments, we use the walking distances from the residential buildings to the candidate locations as a measure.

A3: Focus on locational decisions. The focus of the decision maker is on locational decisions, and routing aspects are considered to better support such decisions. This means that the tours dictated by the solution might not be the ones operated in practice.

A4: Design of collection points. Collection points are designated physical areas where residents must leave their waste bags. There are no bins or containers and bags are placed directly on the ground. Notice that we do not assume a capacity for the candidate locations. If a collection point is assigned to multiple demand nodes and a considerable demand is associated with it, we assume that it is up to the decision maker to determine how to address this at a post-processing stage (e.g., dimensioning the collection point accordingly, visiting the collection point multiple times within the considered time horizon).

A5: Split collection. Due to the nature of a collection point, it is possible to split its waste by gathering some of the bags placed there. This situation is typically encountered at the end of the tour (i.e., before going back to the disposal facility), as the vehicle can only accommodate some of the bags before reaching its capacity. We do not enforce splits to happen at the end of the tours. Since we assume a deterministic demand (A1), they could in principle be planned beforehand at any point in the tour. Furthermore, the routing in this problem is resolved to make better decisions on the location of collection points (A3), and we do not expect the decision maker to operate the tours as determined by the solution. We assume that the collection is scheduled in a way that no more than one vehicle tour will visit the same collection point at the same time.

A6: Stop penalty value. We define the stop penalty value t^{sto} as the time it takes for the collection vehicle to break to reach the collection point, to gather the waste and to accelerate to resume the tour. We assume a constant time per unit of waste to be collected. As the total amount of waste is known (A1), the total time to collect them is constant to the optimization problem, and therefore can be ignored. Thus, the stop penalty value t^{sto} directly represents the break and acceleration time per collection point. For the experiments performed in Section 6, we assume $t^{\text{sto}} = 5$ s.

4. Compact MILP formulations

We propose a compact MILP formulation for the Cm-CTP-R that relies on a road-network graph (Section 4.1). For each tour, integer decision variables capture the number of times that the vehicle traverses a road segment. For comparison purposes, we also characterize a formulation that considers the customer-graph representation of the road network (Section 4.2). In this case, binary decision variables for each pair of potential stopping nodes determine whether or not a vehicle successively stops at both nodes.

4.1. Road-network-based formulation

Let $\mathcal{M} = \{1, \dots, m\}$ be the set of tours enumerated from 1 to m. For each arc $(h, h') \in A$ and tour $k \in \mathcal{M}$, we introduce an integer variable $x_{hh'k}$ that indicates the number of traversals of tour k on arc (h, h'). For each node $j \in V^{\text{sto}}$ and tour $k \in \mathcal{M}$, we define a binary variable y_{jk} that takes value 1 if in tour k the vehicle stops at node j. For each node $i \in W$ and node $j \in V_i^{\text{rank}}$, we add a binary variable z_{ij} that takes value 1 if a demand of node i is satisfied at node j. For each node $j \in V^{\text{sto}}$ and tour $k \in \mathcal{M}$, we introduce a non-negative continuous variable q_{jk} that indicates the quantity of demand satisfied at node j in tour k. For the variables x to define a tour, we adapt the single-commodity flow-based formulation proposed by Letchford et al. (2013) for the STSP by introducing a non-negative continuous variable $f_{hh'k}$ for each tour $k \in \mathcal{M}$ to capture the flow passing through $(h, h') \in A$.

We construct the road-network-based formulation with splits (RN for short) as follows:

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

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

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

$$\sum_{i\in W: j\in V_i^{\text{rank}}} d_i z_{ij} = \sum_{k\in\mathcal{M}} q_{jk} \qquad \forall j \in V^{\text{sto}}, \tag{1d}$$

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

$$\sum_{h\in V:(h,j)\in A} x_{hjk} \ge y_{jk} \qquad \forall j \in V^{\text{sto}}, k \in \mathcal{M}, \tag{1f}$$

$$\sum_{h'\in V:(h',h)\in A} x_{h'hk} - \sum_{h'\in V:(h,h')\in A} x_{hh'k} = 0 \qquad \forall h \in V, k \in \mathcal{M}, \tag{1g}$$

$$\sum_{h'\in V:(h,h')\in A} f_{hh'k} - \sum_{h'\in V:(h',h)\in A} f_{h'hk} = \begin{cases} q_{hk}, & \forall h \in V^{\text{sto}} \\ 0, & \forall h \in V \setminus (V^{\text{sto}} \cup \{\sigma\}) \end{cases} \qquad \forall k \in \mathcal{M}, \tag{1h}$$

$$\sum_{h\in V:(h,\sigma)\in A} f_{h\sigma k} = \sum_{j\in V^{\text{sto}}} q_{jk} \qquad \forall k \in \mathcal{M}, \tag{1i}$$

$$x_{hh'k} \in \mathbb{Z}_{\geq 0} \qquad \forall (h,h') \in A, k \in \mathcal{M}, \tag{1j}$$

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

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

$$0 \leq q_{jk} \leq Q y_{jk} \qquad \forall j \in V^{\text{sto}}, k \in \mathcal{M}, \tag{1m}$$

$$0 \leq f_{hh'k} \leq Q x_{hh'k} \qquad \forall (h,h') \in A, k \in \mathcal{M}. \tag{1n}$$

The objective function (1a) expresses the total cost, which is computed as the sum of the total travel distances 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 stopping node from its rank. Constraints (1c) are the operational constraints which state that the demand of node $i \in W$ is satisfied at the first stopping node in V_i^{rank} at which the vehicle stops. Constraints (1d) guarantee that the total demand that is satisfied at node $j \in V^{\text{sto}}$ is equal to the total quantity that is satisfied in all tours in which the vehicle stops at this node. Constraints (1e) limit the demand satisfied in tour k to the vehicle capacity Q.

Constraints (1f) to (1n) force the variables x to take values that make up valid tours. More precisely, constraints (1f) specify that in tour k the vehicle can only stop at node $j \in V^{\text{sto}}$ if it traverses an incoming arc into node j at least once. The degree constraints (1g) define that in tour k the vehicle enters and leaves any node $h \in V$ the same number of times. Constraints (1h) ensure that the net outflow out of any stopping node $j \in V^{\text{sto}}$ must be q_{jk}, which is the demand quantity satisfied at node j in tour k. For any other node, these constraints impose a 0 net outflow. Constraints (1i) enforce that the total quantity that must go into the depot node equals the total quantity satisfied in tour k.

Finally, constraints (1j) to (1n) define the domain of the decision variables. Additionally, constraints (1m) link the variables q with the variables y by stating that a positive quantity can be satisfied at node $j \in V^{\text{sto}}$ in tour k only if the vehicle stops at this node. Note that Q is a trivial upper bound on the quantity that can be satisfied at any node in one tour. Constraints (1n) link the variables f with the variables x. If $x_{hh'k} = 0$, i.e., arc (h,h') is not traversed in tour k, then the flow $f_{hh'k}$ does not pass through it, and therefore $f_{hh'k} = 0$. If $x_{hh'k} = 1$, the constraint is trivially satisfied as the total flow cannot be larger than Q.

We briefly discuss how the tour $k \in \mathcal{M}$ is constructed with the variables x. To this end, we build an auxiliary directed multigraph $G_k^{\text{T}} = (V_k^{\text{T}}, A_k^{\text{T}})$, where $V_k^{\text{T}} \subseteq V$. We add a node $h \in V$ to V_k^{T} if an arc incident to h, i.e., (h,h') or (h',h), has a positive value $x_{hh'k}$ or $x_{h'hk}$. For any arc $(h,h') \in A$, we add $x_{hh'k}$ copies of (h,h') to A_k^{T}. We then search for an Eulerian cycle in G_k^{T}. Clearly, not all graphs admit an Eulerian cycle. A directed multigraph D admits an Eulerian cycle if and only if D is connected and the in-degree equals the out-degree at each node of D (see, e.g., Bang-Jensen & Gutin, 2008, Section 1.6). The degree condition is directly specified on the variables x in constraint (1g). The connectedness condition is ensured thanks to constraints (1h) and (1i). Indeed, the variable $f_{hh'k}$ can only be positive if $x_{hh'k} \geq 1$. This ensures that all stopping nodes with positive demands visited in tour k are connected to the depot in G_k^{T}. Thus, they are all in the same connected component of G_k^{T}. We can efficiently find an Eulerian cycle in this component, for example, with Hierholzer's algorithm. Note that in suboptimal solutions, we may have other connected components in G_k^{T}. These reflect unnecessary traversals and can simply be deleted. These changes can only improve the quality of the solution.

As shown in Archetti et al. (2006), if the cost matrix satisfies the triangle inequality, then there exists an optimal solution to the SDVRP where the number of splits is less than the number of tours. The number of splits is the sum of the number of splits at each customer, that is defined as the number of tours that visit the customer minus one. We can derive a valid inequality to RN from this property with respect to the potential stopping nodes actually visited in the tours. This prevents to obtain solutions with many unnecessary splits early in the optimization process. To calculate the number of splits, we introduce the binary variable s_j for each node $j \in V^{\text{sto}}$, which takes value 1 if node j is visited at least once. This allows not to take into account the potential stopping nodes that are not visited in any tour. These variables are linearly characterized by the following constraints:

$$s_j \leq \sum_{k\in\mathcal{M}} y_{jk} \qquad \forall j \in V^{\text{sto}}, \tag{2a}$$

$$s_j \geq y_{jk} \qquad \forall j \in V^{\text{sto}}, k \in \mathcal{M}. \tag{2b}$$

Constraints (2a) force s_j to be equal to 0 if node j is not visited, whereas constraints (2b) set s_j to 1 if the node is visited in at least one tour. The valid inequality is then written as follows:

$$\sum_{j\in V^{\text{sto}}} \left(\sum_{k\in\mathcal{M}} y_{jk} - s_j \right) \leq m - 1. \tag{2c}$$

Note that when node $j \in V^{\text{sto}}$ is not visited in any tour, the term in brackets in the left-hand side of constraint (2c) is equal to 0. If the node is visited at least once, this term corresponds to the number of splits, i.e., number of tours that visit the node minus one.

4.2. Customer-based-graph formulation

As discussed in Section 2, VRPs are typically formulated using a so-called customer-based graph. In this section, we show how to construct a customer-based-graph formulation for the Cm-CTP-R, called CG for short, by adapting RN.

To derive CG from RN, we only need to replace V with V' and A with A' in formulation (1). In the CG formulation (4), the variables $x_{jj'k}$ determine whether in tour k the vehicle stops at node j' right after stopping at node j. We can strengthen constraints (1f). In the customer-based graph, if a tour k goes through node j, then it also stops there, whereas in the road-network graph it could just pass by without stopping. Thus, we can replace constraints (1f) with the following stronger condition:

$$\sum_{j'\in V'} x_{j'jk} = y_{jk}, \quad \forall j \in V^{\text{sto}}, k \in \mathcal{M}. \tag{3}$$

The remaining constraints are equivalent to the corresponding ones in RN. Notice that constraints (2) are also valid inequalities to CG.

$$\min \quad \sum_{k\in\mathcal{M}} \sum_{(j,j')\in A'} c_{jj'} x_{jj'k} + \sum_{k\in\mathcal{M}} \sum_{j\in V^{\text{sto}}} t^{\text{sto}} y_{jk}, \tag{4a}$$

$$\text{s.t.} \quad \sum_{j\in V_i^{\text{rank}}} z_{ij} = 1 \qquad \forall i \in W, \tag{4b}$$

$$\sum_{\substack{j'\in V_i^{\text{rank}}:\\ \text{rank}(i,j')>\text{rank}(i,j)}} z_{ij'} \leq 1 - y_{jk} \qquad \forall i \in W, j \in V_i^{\text{rank}}, k \in \mathcal{M}, \tag{4c}$$

$$\sum_{i\in W: j\in V_i^{\text{rank}}} d_i z_{ij} = \sum_{k\in\mathcal{M}} q_{jk} \qquad \forall j \in V^{\text{sto}}, \tag{4d}$$

$$\sum_{j\in V^{\text{sto}}} q_{jk} \leq Q \qquad \forall k \in \mathcal{M}, \tag{4e}$$

$$\sum_{j'\in V'} x_{j'jk} = y_{jk} \qquad \forall j \in V^{\text{sto}}, k \in \mathcal{M}, \tag{4f}$$

$$\sum_{j'\in V'} x_{j'jk} - \sum_{j'\in V'} x_{jj'k} = 0 \qquad \forall j \in V', k \in \mathcal{M}, \tag{4g}$$

$$\sum_{j'\in V^{\text{sto}}} f_{jj'k} - \sum_{j'\in V^{\text{sto}}} f_{j'jk} = q_{jk}, \qquad \forall j \in V^{\text{sto}}, k \in \mathcal{M}, \tag{4h}$$

$$\sum_{j'\in V'} f_{j'\sigma k} = \sum_{j\in V^{\text{sto}}} q_{jk} \qquad \forall k \in \mathcal{M}, \tag{4i}$$

$$x_{jj'k} \in \mathbb{Z}_{\geq 0} \qquad \forall j,j' \in V', k \in \mathcal{M}, \tag{4j}$$

$$y_{jk} \in \{0,1\} \qquad \forall j \in V^{\text{sto}}, k \in \mathcal{M}, \tag{4k}$$

$$z_{ij} \in \{0,1\} \qquad \forall i \in W, j \in V_i^{\text{rank}}, \tag{4l}$$

$$0 \le q_{jk} \le Q y_{jk} \qquad \forall j \in V^{\text{sto}}, k \in \mathcal{M}, \tag{4m}$$

$$0 \le f_{jj'k} \le Q x_{jj'k} \qquad \forall j, j' \in V', k \in \mathcal{M}. \tag{4n}$$

5. A two-phased heuristic method for the C*m*-CTP-R

The compact formulations developed in Section 4 become very large for instances of relevant size, which makes general-purpose MILP solvers to fail at finding solutions. In this section, we propose a two-phased heuristic method where each phase addresses each of the two interdependent subproblems the C*m*-CTP-R is built on. Let us consider the complete directed graph that provides a customer-based representation of the road network $G' = (V', A')$. Recall that $V' = \{\sigma\} \cup V^{\text{sto}}$, where σ denotes the depot and V^{sto} the set of potential stopping nodes.

The first phase handles the generation of subsets of V^{sto} such that all demand nodes are covered. We call such a subset a *set cover* and denote it by $V^{\text{sel}} \subseteq V^{\text{sto}}$. Note that this subproblem is a SCP. Given a set cover V^{sel}, the tours that visit its nodes are built in the second phase. This subproblem is a SDVRP, as the same node might be visited in multiple tours (due to the split feature). The order in which the set covers are processed in the second phase is determined by their cost. This cost provides an estimation of the routing costs associated with a set cover. It is defined as the cost of a giant tour (also known as TSP tour) that starts and ends at the depot and either visits a stopping node of the set cover or one of its alternatives. An alternative of a stopping node is another stopping node that can cover at least the same demand nodes. The set of alternatives to the stopping nodes in V^{sel} is denoted by $V^{\text{alt}} \subseteq V^{\text{sto}}$. We identify these nodes with the goal of increasing the flexibility in the routing and taking into account the road-network structure. For instance, alternative nodes that represent nodes across the street might bring about an improvement in the routing.

A pseudocode of the overall method can be found in Algorithm 1. For the sake of clarity, Algorithms 2 and 3 include the instructions performed in the first and second phase, respectively. Both phases are run in parallel thanks to a multi-threaded implementation of the method. More precisely, one thread is dedicated to the first phase and the remaining ones to the second phase, as building tours requires a higher computational effort. To enable communication between the parallel threads so that the two tasks (i.e., first and second phase) are executed when needed, and to keep track of the set covers that have already been processed in the second phase, we introduce two synchronized collections of set covers named bestSetCovers and treatedSetCovers. Furthermore, we keep track of the solution with the lowest overall cost named bestSol.

The data structures bestSetCovers and treatedSetCovers are shared by the different threads and work as follows. The set covers generated in the first phase are stored in bestSetCovers, which is a min–max priority queue that sorts the set covers according to their cost with maxLength as its maximum length. This structure provides an order to process the set covers in the second phase and allows to efficiently access its minimum or maximum element. When the algorithm starts, bestSetCovers is empty, which triggers the thread dedicated to the first phase to generate new set covers. The remaining threads are waiting until a set cover is available in the queue, which is then retrieved from the queue to be treated in the second phase. As soon as the queue becomes *almost empty* (i.e., its size is less than maxLength), the first thread starts generating new set covers again, so that the threads dedicated to the second phase do not need to wait for new set covers. If a set cover has already been processed in the second phase, we add it to treatedSetCovers.

In the first phase (Algorithm 2), we iteratively generate set covers and calculate their cost with the procedure constructSet() (see Section 5.1). If a set cover V^{sel} belongs to treatedSetCovers, we penalize this fact so that this set cover will only be processed again if there are no other available unprocessed set covers. The set cover V^{sel} is added to bestSetCovers if it is different from the ones already included in bestSetCovers (i.e., it does not contain the same nodes) and either (i) its cost is lower than cost(bestSetCovers.max()) (the largest cost in bestSetCovers) or (ii) the length of bestSetCovers is lower than maxLength.

The goal of the second phase (Algorithm 3) is to solve a SDVRP on the set covers in bestSetCovers. Once a set cover V^{sel} is selected, it is added to treatedSetCovers so that the feedback mechanism (i.e., penalization) with the first phase can be applied. We first transform the SDVRP into a CVRP by means of an a priori splitting strategy of the nodes (see Section 5.2). Second, we use a state-of-the-art algorithm (namely HGS-CVRP) to solve the resulting CVRP (see Section 5.3). This solution is then transformed into a solution to the original SDVRP and compared against the current best solution (bestSol) with respect to their cost. Note that *cost*() is a function which is computed differently for set covers (i.e., cost of giant tours) and solutions (i.e., cost of SDVRP solutions).

Algorithm 1: Two-phased heuristic method for the C*m*-CTP-R

Input: $G' = (V', A')$ directed, complete and strongly connected graph, demand nodes W, d_i and V_i^{rank} $\forall i \in W$, p threads
Output: Solution to the C*m*-CTP-R

```
Define the data structures shared by the different threads:
    bestSetCovers: empty min-max priority queue
    treatedSetCovers ← ∅
    bestSol ← ∅
Run Algorithm 2 in thread 1
Run Algorithm 3 in threads 2, ..., p
return bestSol
```

Algorithm 2: First phase in the two-phased heuristic method (Algorithm 1)

```
while stopping criterion do
    if |bestSetCovers| < maxLength then
        Obtain V^sel with constructSet()
        if V^sel ∈ treatedSetCovers then
            Penalize V^sel
        if V^sel ∉ bestSetCovers and (cost(V^sel) < cost(bestSetCovers.max())
          or |bestSetCovers| < maxLength) then
            bestSetCovers ← bestSetCovers ∪ V^sel
    else
        Wait until the second phase analyzes more set covers, i.e.,
          until |bestSetCovers| < maxLength (Algorithm 3)
```

Algorithm 3: Second phase in the two-phased heuristic method (Algorithm 1)

```
while stopping criterion do
    if bestSetCovers ≠ ∅ then
        V^sel ← bestSetCovers.min() ;
        bestSetCovers ← bestSetCovers \ V^sel ;
        treatedSetCovers ← treatedSetCovers ∪ V^sel ;
        Transform SDVRP associated with V^sel into a CVRP;
        Find solution of the CVRP with HGS-CVRP: CVRPSol ;
        Transform CVRPSol into a SDVRP solution: SDVRPSol ;
        if cost(SDVRPSol) < cost(bestSol) then
            bestSol ← SDVRPSol;
    else
        Wait until additional set covers are generated in the first
          phase, i.e., until bestSetCovers ≠ ∅ (Algorithm 2)
```

Both phases are run until the *stopping criterion* is met. In the computational experiments (see Section 6.3), we set a 3-hour time limit (TL) and terminate the heuristic method after 100 iterations without improvement, where one iteration corresponds to the processing of one set cover, or after TL at the latest.

5.1. Set construction

The procedure constructSet() generates a set cover $V^{sel} \subseteq V^{sto}$ by iteratively adding nodes until the resulting set is a set cover. It then looks for redundant nodes, that is, nodes that can be removed from V^{sel} while it continues to be a set cover. The set of alternatives $V^{alt} \subseteq V^{sto}$ is obtained by identifying the alternatives to each node in V^{sel}. Finally, we define the cost of a set cover as the cost of the giant tour on $V^{sel} \cup V^{alt} \subseteq V^{sto}$. A pseudocode of this procedure is shown in Algorithm 4.

We denote by $W^{cov} \subseteq W$ the set of demand nodes covered by the stopping nodes in V^{sel}. Both W^{cov} and V^{sel} are initially empty. We randomly select a demand node $i \in W \setminus W^{cov}$. To cover this node, we add one of the stopping nodes in V_i^{rank} to V^{sel}. To intensify the covering of demand nodes, for each stopping node $j \in V_i^{rank}$, we calculate the total number of demand nodes that are covered if j is added to V^{sel}. Then, we select and add to V^{sel} a node maximizing this value. We update the set of covered demand nodes by adding all the nodes covered by j, i.e., $W^{cov} = W^{cov} \cup W_j$, where $W_j \subseteq W$ is the set of nodes whose demand is covered by j, i.e., $W_j = \{i \in W | j \in V_i^{rank}\}$. Once V^{sel} is constructed, we remove its redundant nodes. A node $j \in V^{sel}$ is redundant if $V^{sel} \setminus \{j\}$ is a set cover.

To generate V^{alt}, let $\overline{W}_j \subseteq W$ be the set of nodes whose demand is satisfied at $j \in V^{sel}$. It contains all demand nodes i for which j is the first potential stopping node in V_i^{rank} among the ones in V^{sel}, i.e., for any other node $j' \in V_i^{rank} \cap V^{sel}$, $\text{rank}(i,j) < \text{rank}(i,j')$. Let $V_j^{alt} \subseteq V^{sto}$ be the set of alternatives to $j \in V^{sel}$. It contains the nodes $j' \in V^{sto} \setminus \{j\}$ that can cover the demand nodes in $\overline{W}_j$. The set of alternatives is then defined as $V^{alt} = \cup_{j \in V^{sel}} V_j^{alt}$.

The last step consists in building a giant tour using the nodes in $V^{sel} \cup V^{alt}$. The construction of such tour is based on the savings obtained from the gradual insertion of stopping nodes (inspired by Clarke & Wright, 1964). We denote by $g_j \subseteq V^{sto}$ the *group* of potential stopping nodes defined by the set cover node j and its alternatives, i.e., $g_j = \{j\} \cup V_j^{alt}$. To start the giant tour, a node $j \in V^{sel}$ is randomly selected and added to the tour. The cost of the tour (σ, j, σ) is $\ell_{\sigma j} + \ell_{j\sigma}$, where $\ell_{\sigma j}$ and $\ell_{j\sigma}$ respectively denote the shortest path from σ to j and vice versa. The next nodes are gradually inserted either before the first node or after the last node in the current tour. This is also decided at random. Note that the first node added to the tour is both considered the first and the last node. Once the position is resolved, the node to be inserted is the one reporting the largest savings among the nodes belonging to available groups (see Fig. 2). A group is no longer available if any of its nodes has been inserted in the tour. We define an auxiliary set to keep track of the available groups. We denote it by G^{avail} and initialize it to contain all the groups, i.e., $G^{avail} = \cup_{j \in V^{sel}} g_j$. For each node j' added to the giant tour, G^{avail} is updated by removing the groups that contain j', i.e., $G^{avail} = G^{avail} \setminus \{g \in G^{avail} | j' \in g\}$. The insertion of nodes in the tour is repeated until all groups are unavailable, i.e., until G^{avail} is empty. The set cover V^{sel} is then redefined with the stopping nodes that are visited in the giant tour.

Fig. 2 illustrates the calculation of savings. Imagine the next node is to be inserted after the last node in the giant tour, which is denoted by j (Fig. 2(a)). The group $g_{j'}$ comprises the set cover node j' (black node) and its alternatives (white nodes). The savings obtained from inserting a node j'' (gray node) after j are calculated with respect to the shortest path that connects the depot with the group j'' belongs to ($g_{j'}$) back and forth, which is denoted by $\ell_{g_{j'}}$ and is defined as $\min_{z \in g_{j'}} \{\ell_{\sigma z} + \ell_{z\sigma}\}$. More precisely, the savings are defined as $s_{jj''} = -\ell_{g_{j'}} - \ell_{j\sigma} + \ell_{jj''} + \ell_{j''\sigma}$, where $\ell_{g_{j'}}$ can be saved from being included in the giant tour (hence the negative sign), $\ell_{j\sigma}$ is the link removed from the giant tour and $\ell_{jj''}$ and $\ell_{j''\sigma}$ are the links added to the tour. We calculate the savings for all nodes not yet in the giant tour and finally insert the one that maximizes the savings with respect to j, i.e., $j^* = \arg\min_{j' \in g, g \in G^{avail}} s_{jj'}$.

Algorithm 4: constructSet()

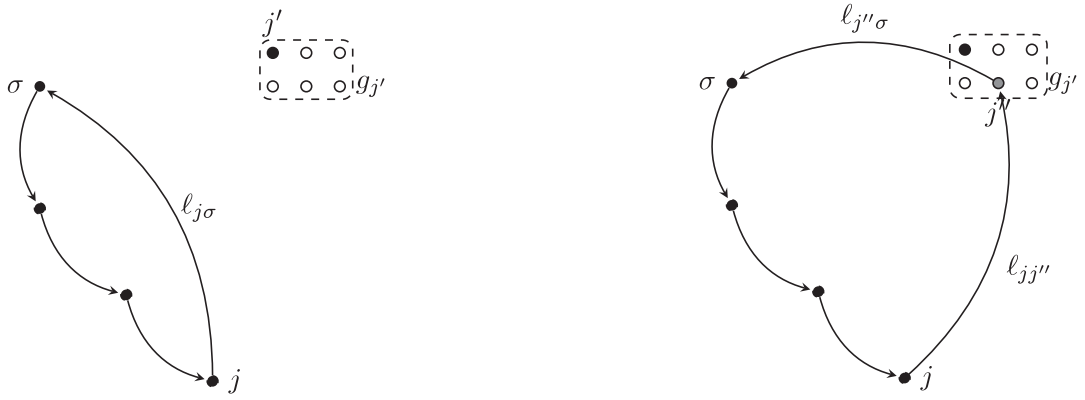
Input: $V' = \{\sigma\} \cup V^{sto}$, W, $V_i^{rank}\ \forall i \in W$
Output: Set cover and its associated cost

Define $V^{sel} \leftarrow \emptyset, W^{cov} \leftarrow \emptyset$
while *V^{sel} is not a set cover* **do**
  Randomly select $i \in W \setminus W^{cov}$
  Select $j^* = \arg\max_{j \in V_i^{rank}} |W^{cov} \cup W_j|$, with $W_j = \{i' \in W | j \in V_{i'}^{rank}\}$
  $V^{sel} \leftarrow V^{sel} \cup \{j^*\}$
  $W^{cov} \leftarrow W^{cov} \cup W_{j^*}$
for $j \in V^{sel}$ **do**
  if *$V^{sel} \setminus \{j\}$ is a set cover* **then**
    $V^{sel} \leftarrow V^{sel} \setminus \{j\}$
for $j \in V^{sel}$ **do**
  Define $\overline{W}_j = \{i \in W | j = \arg\min_{j' \in V_i^{rank}} \text{rank}(i, j')\}$
  Define $V_j^{alt} = \{j' \in V^{sto} \setminus \{j\} \mid \overline{W}_j \subseteq W_{j'}\}$
Define $g_j = \{j\} \cup V_j^{alt}, \forall j \in V^{sel}, G^{avail} = \cup_{j \in V^{sel}} g_j$
Randomly select $j \in V^{sel}$
Define giantTour $\leftarrow (\sigma, j, \sigma)$, cost(giantTour) $\leftarrow \ell_{\sigma j} + \ell_{j\sigma}$
Update $G^{avail} \leftarrow G^{avail} \setminus \{g \in G^{avail} | j \in g\}$
while *G^{avail} is not empty* **do**
  Randomly select the position in the tour to insert the next node: before or after
  if *before* **then**
    Denote by j the first node in the giant tour
    Calculate $s_{j'j} = -\ell_g - \ell_{\sigma j} + \ell_{\sigma j'} + \ell_{j'j}$, where $\ell_g = \arg\min_{z \in g} \{\ell_{\sigma z} + \ell_{z\sigma}\}, \forall j' \in g, g \in G^{avail}$
    Select $j^* = \arg\min_{j' \in g, g \in G^{avail}} s_{j'j}$
    Add j^* to giantTour
    Update cost(giantTour) $\leftarrow$ cost(giantTour) $+ s_{j^*j}$, $G^{avail} \leftarrow G^{avail} \setminus \{g \in G^{avail} | j^* \in g\}$
  if *after* **then**
    Denote by j the last node in the giant tour
    Calculate $s_{jj'} = -\ell_g - \ell_{j\sigma} + \ell_{jj'} + \ell_{j'\sigma}$, where $\ell_g = \arg\min_{z \in g} \{\ell_{\sigma z} + \ell_{z\sigma}\}, \forall j' \in g, g \in G^{avail}$
    Select $j^* = \arg\min_{j' \in g, g \in G^{avail}} s_{jj'}$
    Add j^* to giantTour
    Update cost(giantTour) $\leftarrow$ cost(giantTour) $+ s_{jj^*}$, $G^{avail} \leftarrow G^{avail} \setminus \{g \in G^{avail} | j^* \in g\}$
Define cost(V^{sel}) = cost(giantTour)
return *V^{sel}, cost(V^{sel})*

5.2. From SDVRP to CVRP

To solve the SDVRP associated with a set cover V^{sel}, we consider the method developed by Chen et al. (2017). The idea is to transform the SDVRP into a CVRP at the expense of an increased number of customers. To do so, an a priori splitting strategy is used, i.e., each customer demand is split in advance and not as determined by the solution of the problem. This approach allows to use any CVRP solver instead of developing tailored algorithms for the SDVRP. Finally, the solution to the CVRP is transformed into a solution to the original SDVRP.

Chen et al. (2017) propose a 20/10/5/1 rule that splits each customer demand into m_{20} pieces of $0.2Q$, m_{10} pieces of $0.1Q$, m_5 pieces of $0.05Q$, m_1 pieces of $0.01Q$, and at most one remaining piece of less than $0.01Q$, where Q is the vehicle capacity. The number of pieces for each customer j is calculated as follows: $m_{20} = \max\{m \in \mathbb{Z}^+ \cup \{0\} | 0.2Qm \leq d_j\}$, $m_{10} = \max\{m \in \mathbb{Z}^+ \cup \{0\} | 0.1Qm \leq d_j - 0.2Qm_{20}\}$, $m_5 = \max\{m \in \mathbb{Z}^+ \cup \{0\} | 0.05Qm \leq d_j - 0.2Qm_{20} - 0.1Qm_{10}\}$ and $m_1 = \max\{m \in \mathbb{Z}^+ \cup \{0\} | 0.05Qm \leq d_j - 0.2Qm_{20} - 0.1Qm_{10} - 0.05Qm_5\}$, where d_j is the demand of customer j. For instance, if $Q = 100$ and $d_j = 76$, this customer is split into six nodes with demands 20, 20, 20, 10, 5, and 1. Notice that the only way not to eliminate the optimal SDVRP



(a) Current giant tour (j denotes the last visited node) and group of potential stopping nodes $g_{j'}$.

(b) Savings obtained from inserting node $j'' \in g_{j'}$ after node j: $s_{jj''} = -\ell_{g_{j'}} - \ell_{j\sigma} + \ell_{jj''} + \ell_{j''\sigma}$.

Fig. 2. Calculation of the savings for the insertion of a group of potential stopping nodes after the last node of the current giant tour.

solution when both the vehicle capacity and the customer demands are integers is to split the demands into unit demands. Nevertheless, this strategy will dramatically increase the size of the resulting CVRP, and consequently the running time. As pointed out by the authors, the split rule might not eliminate the optimal solution if the vehicles (tours in our case) that serve split demands are not fully loaded, since this involves many different ways to split the demand in an optimal solution.

For the Cm-CTP-R, the amount of demand to be satisfied at each stopping node is determined during the first phase, when the stopping nodes that form the set cover are decided. We then only split the nodes whose demand exceeds 10% of the vehicle capacity, i.e., the set cover nodes j such that $d_j \geq 0.1Q$, with the 20/10/5/1 split rule. In this way, we prevent to split small demands (with respect to the vehicle capacity), and therefore limit the number of nodes that are generated when defining the associated CVRP.

5.3. Hybrid genetic search for the CVRP

We rely on the hybrid genetic search for the CVRP (HGS-CVRP) developed by Vidal (2022). This state-of-the-art algorithm uses the same general methodology as Vidal et al. (2012) but includes an additional neighborhood called *SWAP** that consists in exchanging two customers between different routes without an insertion in place. Computational experiments have shown that HGS-CVRP stands as the leading metaheuristic regarding solution quality and convergence speed.

The performance of HGS-CVRP comes from the combination of three main strategies. First, a synergistic combination of crossover-based and neighborhood-based search. The former allows a diversified search in the solution space, while the latter enables aggressive solution improvement. Second, a controlled exploration of infeasible solutions in which any excess load in the routes is linearly penalized. Third, population diversity management strategies that allow to maintain a diversified and high-quality set of solutions and counterbalance the loss of diversity due to the neighborhood search. We refer the reader to Vidal (2022) for further details on the HGS-CVRP algorithm.

We apply the open-source implementation of HGS-CVRP [2] with a lower number of iterations without improvement (10,000 instead of 20,000 to speed up the process) and the rest of the parameters set to their default values. The algorithm returns a solution file with the nodes visited in each route and the total cost. We then simply need to transform this solution into a solution to the original SDVRP. To do so, we define the tours of the SDVRP solution as the tours of the CVRP solution except for the split nodes introduced in the CVRP. In the SDVRP tour, only the original nodes are visited, and the demand satisfied in the tour is equal to the sum of the demands of the associated split nodes satisfied in the CVRP tour. Notice that these split nodes are visited in consecutive order in the CVRP tour. Indeed, they are located at the same position, and consequently the distance between two split nodes is equal to 0.

[2] https://github.com/vidalt/HGS-CVRP

6. Computational experiments

In this section, we present the tests conducted on a set of realistic instances inspired by real data from municipalities in Switzerland. In Section 6.1, we describe the datasets and problem instances considered for the experiments. Section 6.2 compares the proposed road-network-based MILP formulation against its customer-based counterpart, Section 6.3 assesses the performance of the heuristic method and Section 6.4 discusses some practical aspects of the introduced waste collection method with respect to the state of practice. Note that in the tables included in Sections 6.2 and 6.3 the best results are highlighted in bold.

6.1. Datasets and problem instances

Each problem instance is generated with a road-network dataset containing information on the nodes and the arcs of the underlying graph and by providing values to the problem parameters. The associated customer-based graphs are constructed according to Section 3.1. The two main parameters that characterize an instance are the maximum walking distance (γ) and the number of tours (m). The former is considered to determine the ranks of each residential building, i.e., the candidate locations within a distance less or equal than γ ordered in increasing distance, as described in Section 3.2. The latter refers to the necessary number of tours to collect all the waste. For the sake of deriving multiple instances, we assume various values for m and derive the vehicle capacity Q accordingly (see Section 3.1). We add a 5% buffer to the minimum required capacity of each vehicle, i.e., $Q = \lceil 1.05 \cdot (d^{\text{tot}}/m) \rceil$. This enables the vehicle not to be fully loaded, which might allow for the a priori splitting of the demand nodes not to eliminate the optimal solution (Chen et al., 2017). We also need to make assumptions on the average speed of the vehicle to transform travel distances into travel times in the objective function. We distinguish between the

Table 1
Characteristics of the road-network graphs for each dataset.

Dataset		*W15*	*W50*	*W100*	*W200*	*W600*
$\|W\|$		15	50	100	200	600
$\|V\|$		154	394	899	1840	8046
$\|A\|$		172	426	1006	2120	8740
Area (km^2)		0.65	2.86	5.06	6.95	19.84
Density ($\|W\|/km^2$)		23.08	17.48	19.76	28.78	30.24
$\|V^{sto}\|$	$\gamma = 0$	23	74	137	263	808
	$\gamma = 50$	45	145	286	439	1489
	$\gamma = 100$	73	177	393	507	1962
	$\gamma = 200$	96	208	474	556	2500
	$\gamma = 300$	114	214	523	584	2861

average speed of the vehicle during collection (s^{col}) and the average speed from or to the disposal facility (s^{dep}).

We define instances of various sizes with respect to the number of demand nodes. In particular, we consider 15, 50, 100, 200, and 600 demand nodes, and label these datasets $W15$, $W50$, $W100$, $W200$ and $W600$, respectively. The smallest dataset ($W15$) is used to solve both MILP formulations to optimality, so that we can compare their performance. The largest dataset ($W600$) is used to assess the performance of the heuristic. We assume $m \in \{1, 2, 6\}$ and $\gamma \in \{50, 100, 200, 300\}$. This results into 12 instances for each dataset, which yields 60 instances in total. They are labeled $Wx-\gamma-m$, with $x \in \{15, 50, 100, 200, 600\}$. For the experiments conducted in Section 6.4 we derive additional instances with $\gamma = 0$ (i.e., no walking distance) to represent the state of practice. Notice that for the instances with $m = 1$, the Cm-CTP-R reduces to a CTP, and for the ones with $m = 1$ and $\gamma = 0$ (i.e., all demand nodes need to be visited), the problem becomes a TSP.

The road-network graph associated with each instance is constructed according to Section 3.2. It comprises the nodes to be covered (i.e., the demand nodes) and the nodes that can be visited, which include both the demand nodes and the nodes representing candidate locations that lie within the assumed maximum walking distance. Table 1 shows the characteristics of the road-network graphs associated with the considered datasets. Notice that the number of candidate locations $|V^{sto}|$ is different for each value of γ, as the ranks are determined according to this value and $V^{sto} = \cup_{i \in W} V_i^{rank}$. Also notice that $|V^{sto}| > |W|$ for $\gamma = 0$ because candidate locations on the other side of the street can be reached at 0 cost (i.e., walking distance), which is why there are more nodes representing candidate locations than demand nodes.

Concerning the other parameters, we assume a larger average speed to go from and to the disposal facility, i.e., $s^{col} = 2$ m/s and $s^{dep} = 14$ m/s. As discussed in Section 3.2, we set the stop penalty value to $t^{sto} = 5$ s, which can be interpreted as the time it takes for a vehicle to break to reach the collection point and to accelerate to resume the tour (the time to collect the waste is constant to the optimization problem, see Section 3.3).

The developed computer codes are implemented in Java. The instances were tested on a computer with a 3.4 GHz Intel Core i5 processor, 32 GB of RAM, operating under Windows 10. To solve the MILP formulations, we use the Gurobi 9.1.2 MIP solver via its Java API.

6.2. Comparison of the MILP formulations

The goal of this section is to test and compare against each other the MILP formulations introduced in Section 4. We set a three-hour TL for both formulations on each instance.

Table 2 presents the results of CG and RN for four of the datasets, namely $W15$, $W50$, $W100$ and $W200$. Note that we exclude dataset $W600$ in this section because of its size. The instances associated with this dataset are tested in Section 6.3. The table shows the total number of instances, the number of instances solved to optimality, the number of instances for which a feasible solution was found (excluding the ones solved to optimality) and the number of instances for which no solution was found. Furthermore, it presents the average and worst gaps reported by Gurobi for the instances for which the 3-hour TL was reached. For a fair comparison of the average computation times, we only consider instances that were solved to optimality by both approaches, namely 7 instances of dataset *W15*.

We observe that for the small instances (datasets $W15$ and $W50$) either an optimal or a feasible solution was found within the TL, whereas for almost half of the larger instances (datasets $W100$ and $W200$) no solution could be found by CG. RN is able to find more often optimal solutions than CG, except for the smallest dataset *W15*, where both formulations could prove optimality to the same number of instances. Over all datasets, RN reported lower average and worst gaps for the instances that were not solved to optimality. For the few instances solved to optimality by both formulations, RN was faster in doing so.

Table 3 presents the disaggregated results for each instance. In particular, it includes the upper bounds, lower bounds, gaps, and the computation times reported by Gurobi. As already anticipated in Table 2, we observe that RN is able to find more optimal and feasible solutions, and it is faster in proving optimality for the instances solved by both formulations. The only instance that CG could solve to optimality and RN could not is *W15-200-6* for which RN, nevertheless, returned the same upper bound and a gap of 0.09%. Furthermore, regarding solution quality, we observe that for all instances RN found solutions with better or equally good upper bound values.

These experiments show the superiority of RN both from a computational and solution quality point of view. We now analyze the impact of the parameters that define the problem instances (m and γ) on the solvability of RN. For the sake of illustration, we only consider the instances for the dataset $W100$. Table 4 reports the computation times (s) for the instances solved to optimality and the gaps reported by Gurobi (%) for the instances for which the 3-hour TL was reached. We clearly observe that as γ and/or m increase, RN either takes more time in finding the optimal solution or is only able to provide a feasible solution, while the associated gap also increases.

6.3. Validation and performance of the heuristic method

This section aims to validate the heuristic method (H) with respect to the instances solved to optimality by RN and to assess its performance for other instances. To this end, we consider the instances associated with datasets $W50$, $W100$, $W200$ and $W600$. Notice that we exclude dataset $W15$ in this section. We set again a 3-hour TL to run RN on the selected instances. We terminate H after 100 iterations without improvement, where one iteration corresponds to the processing of one set cover, or after TL at the latest.

Tables 5 and 6 present the results of RN and H for each instance and include the upper bounds, computation times and the gaps of H with respect to RN (the ones equal to 0 are highlighted in italics). These gaps are computed as the relative difference of the upper bound value obtained with H with respect to the upper bound value reported by RN (i.e., $(UB^{H} - UB^{RN})/UB^{RN}$).

For the validation of H, we rely on the instances solved to optimality by RN. We observe in Table 5 that H found solutions with the same upper bound values as RN for 13 out of 20 instances. In general, RN was able to find the optimal solution faster mainly for CTP instances ($m = 1$), whereas H was faster for m-CTP instances ($m > 1$). For the seven instances where H was not able to find the optimal solutions, it still provides good solutions with optimality gaps below 1.7%. For two of these instances, H returned the final solution faster than RN. Note that the computational time of H depends on the stopping criterion (in this case iterations without improvement). A different definition of the termination criterion might yield different computational times.

We now assess the performance of H with Table 6. We can see that for instances where the 3-hour time limit was reached by RN, H was always faster (with the exception of the instance *W200-200-2*) and

Table 2
Aggregated results of CG and RN formulations for the instances associated with $W15$, $W50$, $W100$ and $W200$ within a 3-hour time limit.

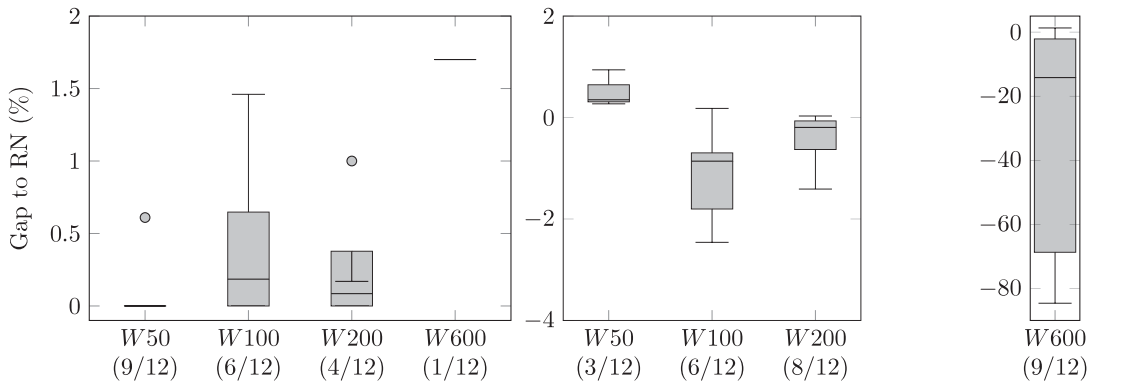
Dataset	*W15*		*W50*		*W100*		*W200*	
	CG	RN	CG	RN	CG	RN	CG	RN
Instances (12 in total)								
# optimal	10	10	1	**9**	0	**6**	0	**4**
# feasible	2	2	11	3	9	6	5	8
# no solution	0	0	0	0	3	**0**	7	**0**
Average gap (%)	0.47	**0.20**	12.34	**1.99**	44.36	**10.67**	42.40	**7.69**
Worst gap (%)	0.51	**0.30**	25.41	**3.87**	60.53	**14.84**	87.08	**13.21**
Average comp. time (s)	1430.90	**311.57**	1449.23	**1.22**	–	–	–	–

Table 3
Exhaustive results of CG and RN formulations for the instances associated with $W15$, $W50$, $W100$ and $W200$ (TL corresponds to a 3-hour time limit).

Instance	Upper bound		Lower bound		Gap (%)		Comp. time (s)	
($Wx-\gamma-m$)	CG	RN	CG	RN	CG	RN	CG	RN
W15-50-1	**6316.65**	**6316.65**	**6316.65**	**6316.65**	**0.00**	**0.00**	4.37	**0.46**
W15-50-2	**9639.65**	**9639.65**	**9639.65**	**9639.65**	**0.00**	**0.00**	296.72	**9.06**
W15-50-6	**23 165.14**	**23 165.14**	23 064.56	**23 165.14**	0.43	**0.00**	TL	**7296.85**
W15-100-1	**6227.43**	**6227.43**	**6227.43**	**6227.43**	**0.00**	**0.00**	536.59	**1.64**
W15-100-2	**9511.86**	**9511.86**	**9511.86**	**9511.86**	**0.00**	**0.00**	4665.50	**92.70**
W15-100-6	**23 017.28**	**23 017.28**	22 936.35	**22 947.86**	0.35	**0.30**	TL	TL
W15-200-1	**5903.72**	**5903.72**	**5903.72**	**5903.72**	**0.00**	**0.00**	81.25	**8.19**
W15-200-2	**9312.28**	**9312.28**	**9312.28**	**9312.28**	**0.00**	**0.00**	4038.74	**194.59**
W15-200-6	**22 842.15**	**22 842.15**	**22 842.15**	22 821.80	**0.00**	0.09	**2935.71**	TL
W15-300-1	**5791.72**	**5791.72**	**5791.72**	**5791.72**	**0.00**	**0.00**	61.15	**8.73**
W15-300-2	**9214.01**	**9214.01**	**9214.01**	**9214.01**	**0.00**	**0.00**	2827.23	**1072.63**
W15-300-6	**22 737.43**	**22 737.43**	**22 737.43**	**22 737.43**	**0.00**	**0.00**	**366.50**	1416.11
W50-50-1	**5524.07**	**5524.07**	**5524.07**	**5524.07**	**0.00**	**0.00**	1449.23	**1.22**
W50-50-2	**7799.64**	**7799.64**	7513.98	**7799.64**	3.66	**0.00**	TL	**104.95**
W50-50-6	17 640.79	**17 602.50**	17 203.21	**17 572.24**	2.48	**0.17**	TL	TL
W50-100-1	**5108.28**	**5108.28**	4938.20	**5108.28**	3.33	**0.00**	TL	**3.83**
W50-100-2	**7114.28**	**7114.28**	6806.97	**7114.28**	4.32	**0.00**	TL	**68.88**
W50-100-6	17 331.70	**17 073.42**	16 387.05	**17 073.42**	5.45	**0.00**	TL	**6765.66**
W50-200-1	**4531.93**	**4531.93**	3967.09	**4531.93**	12.46	**0.00**	TL	**58.84**
W50-200-2	**6728.78**	**6728.78**	5792.34	**6728.78**	13.92	**0.00**	TL	**511.35**
W50-200-6	16 750.00	**16 689.15**	15 853.63	**16 369.51**	5.35	**1.92**	TL	TL
W50-300-1	4100.79	**4095.78**	3363.79	**4095.78**	17.97	**0.00**	TL	**72.41**
W50-300-2	6646.14	**6646.13**	5589.07	**6646.13**	15.90	**0.00**	TL	**1956.12**
W50-300-6	16 812.28	**16 572.14**	15 744.00	**15 930.65**	6.35	**3.87**	TL	TL
W100-50-1	7175.92	**7105.71**	5879.78	**7105.71**	18.06	**0.00**	TL	**8.56**
W100-50-2	9164.71	**8359.57**	6936.61	**8359.57**	24.31	**0.00**	TL	**187.18**
W100-50-6	15 917.92	**15 460.78**	13 069.92	**14 786.74**	17.89	**4.36**	TL	TL
W100-100-1	6188.71	**5839.85**	3817.19	**5839.85**	38.32	**0.00**	TL	**84.78**
W100-100-2	9379.91	**7254.07**	5241.78	**7254.07**	44.12	**0.00**	TL	**8916.96**
W100-100-6	–	**14 545.72**	–	**12 946.90**	–	**10.99**	TL	TL
W100-200-1	5741.13	**5080.64**	3080.23	**5080.64**	46.35	**0.00**	TL	**3685.47**
W100-200-2	8160.49	**6602.71**	4736.13	**6007.99**	41.96	**9.01**	TL	TL
W100-200-6	–	**13 967.06**	–	**11 894.87**	–	**14.84**	TL	TL
W100-300-1	5270.71	**4225.21**	2806.79	**4225.21**	46.75	**0.00**	TL	**8424.55**
W100-300-2	6998.64	**5913.35**	4535.41	**5161.82**	35.20	**12.71**	TL	TL
W100-300-6	–	**13 072.55**	–	**11 491.36**	–	**12.10**	TL	TL
W200-50-1	14 458.93	**14 443.93**	11 982.29	**14 443.93**	17.13	**0.00**	TL	**12.89**
W200-50-2	–	**18 456.36**	–	**18 456.36**	–	**0.00**	TL	**640.75**
W200-50-6	–	**36 392.00**	–	**34 232.78**	–	**5.93**	TL	TL
W200-100-1	12 847.79	**12 350.29**	9515.17	**12 350.29**	25.94	**0.00**	TL	**152.23**
W200-100-2	–	**16 787.85**	–	**16 214.94**	–	**3.41**	TL	TL
W200-100-6	–	**34 918.64**	–	**31 816.50**	–	**8.88**	TL	TL
W200-200-1	13 061.79	**11 006.29**	8264.56	**11 006.29**	36.73	**0.00**	TL	**1186.66**
W200-200-2	–	**15 464.28**	-	**14 393.72**	–	**6.92**	TL	TL
W200-200-6	–	**33 546.71**	–	**30 308.73**	–	**9.65**	TL	TL
W200-300-1	12 308.15	**10 447.22**	8005.84	**10 076.56**	34.96	**3.55**	TL	TL
W200-300-2	80 077.79	**14 990.29**	12 435.27	**13 009.79**	84.47	**13.21**	TL	TL
W200-300-6	–	**33 191.00**	–	**29 883.25**	-	**9.97**	TL	TL

returned better solutions (smaller upper bound values) for most of them (22 out of 28 instances). This effect is also visible in the gaps to RN, which are negative when the upper bound values of H are lower that those of RN. For the instance *W200-200-2*, H did only terminate after the TL was reached, however, it found a better solution than RN with a gap of −0.15%. For the six instances for which RN found a better solution than H, the respective gaps are all below 1.3%.

Fig. 3 shows the gaps to RN for the considered instances in a boxplot chart. Fig. 3(a) visualizes the gaps for the instances solved to optimality by RN (Table 5) and Figs. 3(b) and 3(c) visualize the gaps for the instances for which RN reached the 3-hour TL (Table 6). The value below the dataset names refer to the number of instances included in each of the plots. These plots illustrate in an aggregated way that H was able to find solutions that are optimal or close to optimality for instances that were solved to optimality by RN, and returned better



(a) Solved to optimality (b) TL was reached (c) TL was reached

Fig. 3. Boxplots of the gaps to RN for the instances associated with the datasets $W50$, $W100$, $W200$ and $W600$.

Table 4
Sensitivity analysis with respect to m and γ for the instances associated with *W100* solved by RN (TL corresponds to a 3-hour time limit).

γ/m	Comp. time (s)			Gap (%)		
	1	2	6	1	2	6
50	8.56	187.18	TL	0.00	0.00	4.36
100	84.78	8916.96	TL	0.00	0.00	10.99
200	3685.47	TL	TL	0.00	9.01	14.84
300	8424.55	TL	TL	0.00	12.71	12.10

Table 5
Comparison of results of RN and H for the instances associated with $W50$, $W100$, $W200$ and $W600$ solved to optimality.

Instance	Upper bound		Comp. time (s)		Gap to RN (%)
($Wx-\gamma-m$)	RN	H	RN	H	
W50-50-1	**5524.07**	**5524.07**	**1.22**	36.21	*0.00*
W50-50-2	**7799.64**	**7799.64**	**104.95**	3456.64	*0.00*
W50-100-1	**5108.28**	**5108.29**	**3.83**	24.66	*0.00*
W50-100-2	**7114.28**	**7114.29**	**68.88**	2175.56	*0.00*
W50-100-6	**17 073.42**	17 178.14	6765.66	**216.27**	0.61
W50-200-1	**4531.93**	**4531.93**	58.84	**15.58**	*0.00*
W50-200-2	**6728.78**	**6728.79**	511.35	**72.34**	*0.00*
W50-300-1	**4095.78**	**4095.79**	72.41	**12.39**	*0.00*
W50-300-2	**6646.13**	**6646.14**	1956.12	**172.32**	*0.00*
W100-50-1	**7105.71**	7131.71	**8.56**	2846.55	0.37
W100-50-2	**8359.57**	8421.57	**187.18**	10 182.24	0.74
W100-100-1	**5839.85**	**5839.86**	**84.78**	1212.24	*0.00*
W100-100-2	**7254.07**	**7254.07**	8916.96	**8274.15**	*0.00*
W100-200-1	**5080.64**	**5080.64**	**3685.47**	7076.41	*0.00*
W100-300-1	**4225.21**	4286.71	8424.55	**898.79**	1.46
W200-50-1	**14 443.93**	**14 443.93**	**12.89**	6039.10	*0.00*
W200-50-2	**18 456.36**	18 488.36	**640.75**	5554.41	0.17
W200-100-1	**12 350.29**	12 473.29	**152.23**	5804.01	1.00
W200-200-1	**11 006.29**	**11 006.29**	1186.66	**287.86**	*0.00*
W600-50-1	**37 473.78**	38 109.14	**1428.52**	8902.14	1.70

Table 6
Comparison of results of RN and H for the instances associated with $W50$, $W100$, $W200$ and $W600$ (the RN upper bound corresponds to the feasible solution reported by Gurobi after a 3-hour time limit (TL)).

Instance	Upper bound		Comp. time H (s)	Gap to RN (%)
($Wx-\gamma-m$)	RN	H		
W50-50-6	**17 602.50**	17664.93	**1063.79**	0.35
W50-200-6	**16 689.15**	16734.71	**149.80**	0.27
W50-300-6	**16 572.14**	16728.29	**199.49**	0.94
W100-50-6	15 460.78	**15349.36**	**4853.35**	**−0.72**
W100-100-6	14 545.72	**14187.50**	**1842.73**	**−2.46**
W100-200-2	**6602.71**	6614.50	**1162.32**	0.18
W100-200-6	13 967.06	**13677.36**	**5531.96**	**−2.07**
W100-300-2	5913.35	**5872.50**	**1165.59**	**−0.69**
W100-300-6	13 072.55	**12942.00**	**1334.46**	**−1.00**
W200-50-6	36 392.00	**36358.36**	**2239.42**	**−0.09**
W200-100-2	16 787.85	**16747.57**	**1916.30**	**−0.24**
W200-100-6	34 918.64	**34730.57**	**9604.96**	**−0.54**
W200-200-2	15 464.28	**15440.43**	TL	**−0.15**
W200-200-6	**33 546.71**	33558.36	**4169.61**	0.03
W200-300-1	10 447.22	**10447.21**	**1280.51**	*0.00*
W200-300-2	14 990.29	**14855.57**	**1993.02**	**−0.90**
W200-300-6	33 191.00	**32724.29**	**504.41**	**−1.41**
W600-50-2	40 943.57	**39093.71**	**6340.32**	**−4.52**
W600-50-6	158 262.42	**49519.21**	**13 636.53**	**−68.71**
W600-100-1	**31 961.72**	32373.93	**6987.99**	1.29
W600-100-2	39 379.00	**33547.86**	**5109.05**	**−14.81**
W600-100-6	271 135.30	**43387.43**	**883.70**	**−84.00**
W600-200-1	27 550.43	**26958.93**	**3247.14**	**−2.15**
W600-200-2	33 074.79	**28377.86**	**1168.45**	**−14.20**
W600-200-6	–	**37991.43**	**6371.19**	–
W600-300-1	23 552.71	**23077.29**	**988.81**	**−2.02**
W600-300-2	160 821.78	**24739.57**	**9980.07**	**−84.62**
W600-300-6	–	**35111.36**	**4506.08**	–

solutions for most of the instances for which RN found a feasible solution at the 3-hour TL. The largest gap to RN is −84.62% which corresponds to the instance *W600-300-2*.

Fig. 4 visualizes the evolution of the upper bound values over time for H for some instances of the largest dataset *W600*. Each point in the plot represents a new best solution found by the method. We observe that H finds solutions quickly and reports the best found solution early in the solving process, some of them already in the first 15 min. These results clearly manifest that H provides good solutions in reasonable time and is better suited for real-life applications with instances of relevant size.

6.4. Practical aspects

In this section, we analyze the savings in the total collection time with respect to door-to-door collection (no walking distance). As presented in Section 6.3, H performed well in finding good solutions for all dataset sizes, and is therefore considered as the method to use in practice. To this end, we consider the solution generated by H (with the same termination criterion as the one considered in Section 6.3) for the instances associated with the datasets $W50$, $W100$, $W200$ and $W600$. We then compare it against the heuristic solution of the instance associated with the same dataset and number of tours but with a maximum walking distance of $\gamma = 0$ m.

Fig. 5 displays the savings (as a percentage) with respect to door-to-door collection for each dataset, walking distance ($\gamma > 0$) and number

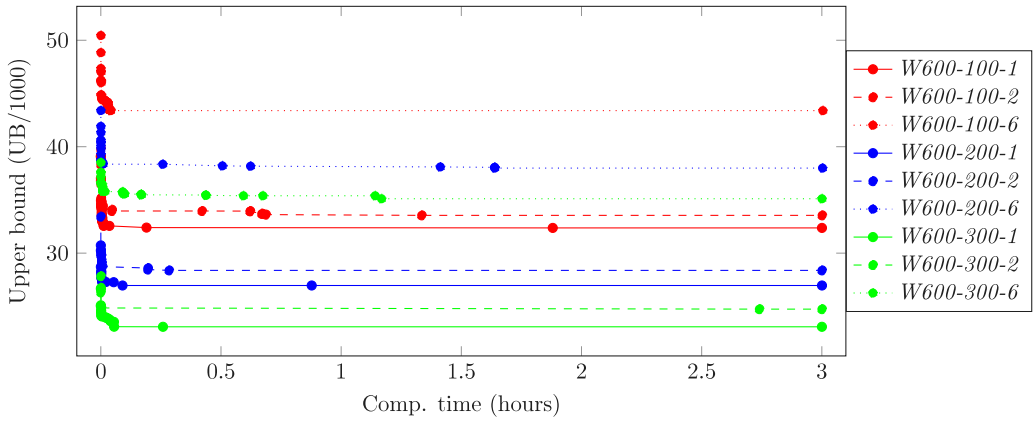
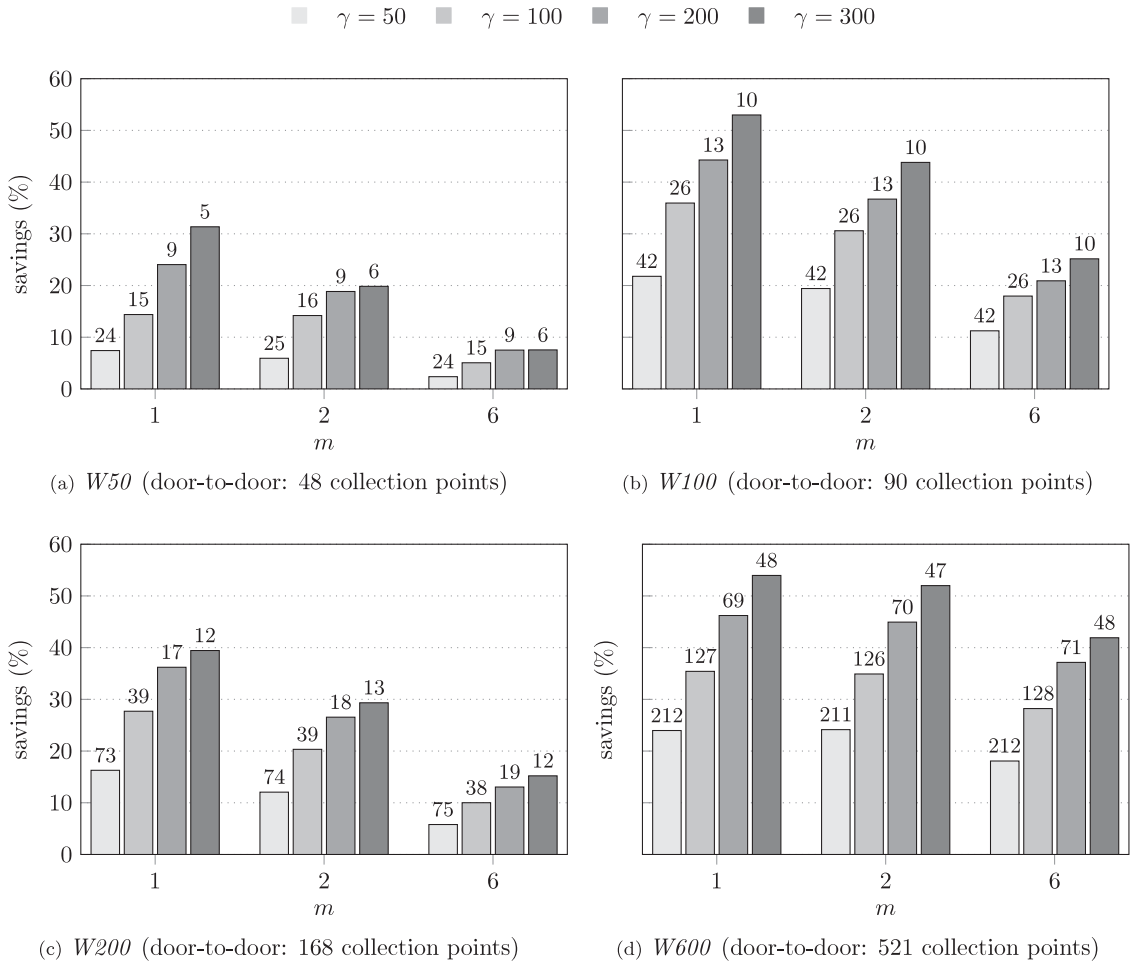


Fig. 4. Convergence graph of H for instances associated with dataset *W600*, walking distances $\gamma = 100$ (red), $\gamma = 200$ (blue), $\gamma = 300$ (green), and number of tours $m \in \{1, 2, 6\}$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) *W50* (door-to-door: 48 collection points)

(b) *W100* (door-to-door: 90 collection points)

(c) *W200* (door-to-door: 168 collection points)

(d) *W600* (door-to-door: 521 collection points)

Fig. 5. Percentage of savings with respect to door-to-door collection ($\gamma = 0$) using the heuristic method to generate the solutions (the labels on top of the bars are the number of collection points visited in the tours).

of tours (m). In all cases, we observe savings of at least 2% (*W50-50-6*) up to a maximum of over 50% (*W100-300-1*) with an average of 25.25%. As expected, the larger the maximum walking distance, the higher the savings. We observe that going from $\gamma = 0$ m to $\gamma = 50$ m comes with a larger gain in collection time (on average 14.04%) than increasing γ further to 100 m, 200 m or 300 m (additional gain with respect to the previous value of γ on average 8.85%, 6.81% and 4.68%, respectively). The same can be observed with the number of collection points visited by the tours (labels on top of the bars). Going from door-to-door collection to $\gamma = 50$ m reduces the number of points by 50% on average while increasing to $\gamma = 100$ m, 200 m or 300 m only removes 40% of the points on average. We also observe that the savings decrease as the value of m increases. This might be due to a larger geographical distance of the network to the disposal facility σ

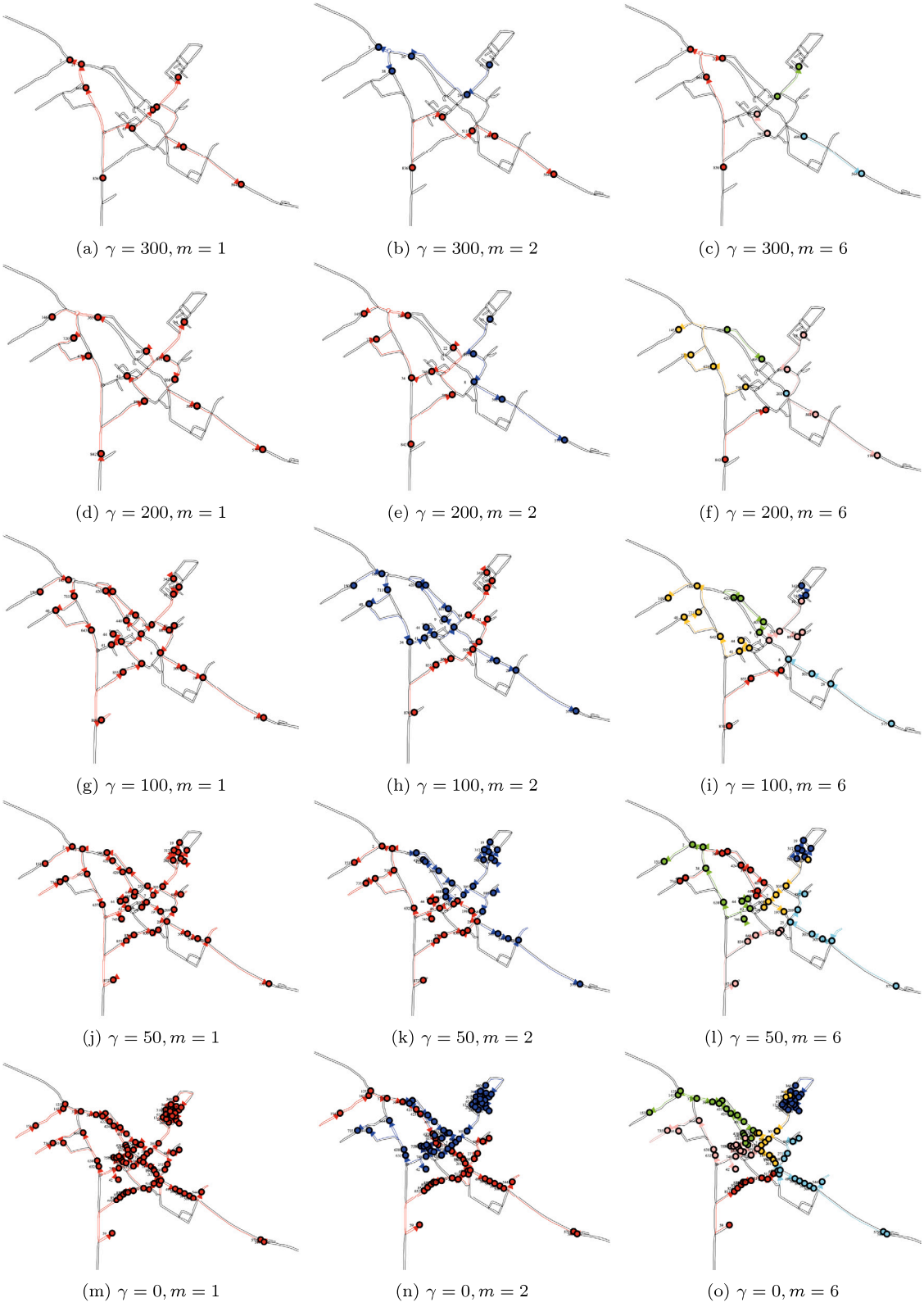


Fig. 6. Solutions of the heuristic approach for all instances of dataset *W100*.

and consequently a higher constant driving time to dump the waste in each tour. To analyze the impact of the number of tours (m) on the locational decision of collection points, we observe that the number of collection points does not change significantly for different values of m. For instance, in Fig. 5(b), for each value of γ (each color), the resulting number of collection points (i.e., label on top of the bars) is the same for all values of m. Taking a closer look at the characteristics of the municipalities (see Table 1), we note that the savings with respect to door-to-door collection are independent of the sizes and densities of the

municipalities. The highest savings could be achieved for *W600*, which is the municipality with the highest number of demand nodes.

To visualize the effect of different maximum walking distances on the number of collection points, Fig. 6 presents the network representation of the solutions obtained by H for all instances (i.e., $\gamma \in \{0, 50, 100, 200, 300\}$ and $m \in \{1, 2, 6\}$) of the dataset *W100*. The tours are marked in different colors, which helps to easily identify the various clusters of collection points. For visualization reasons, the disposal facility is not shown in the images. We observe that with increasing walking distance (images from bottom to top) the number of visited nodes in the graph decreases which, as discussed above, correlates with the gain in collection time. Furthermore, the images show that for each column of instances (same value of m) the clustering of nodes is similar based on the areas in which the nodes lie. For each row of instances (same value of γ), we observe a similar selection of collection points, which supports the above-mentioned claim that the locational decision does not depend on the number of assumed tours.

7. Conclusion

In this paper, we formulated and solved the Cm-CTP-R, a particular version of the m-CTP where the constraints on the length and number of nodes of each tour are replaced by vehicle capacity constraints. Furthermore, the coverage of demand nodes is determined by exogenously given ranks that enforce each demand node to be covered by the first node in its rank which is visited by a vehicle. We developed a road-network-based MILP formulation and compared it against a customer-based formulation as typically used in VRP. To solve practically relevant instances, we proposed a two-phased heuristic that first generates sets of nodes to be visited (set covers) and then determines the tours that visit them. Finally, we derived multiple instances inspired by real-life data from municipalities of Switzerland to perform extensive experiments. Additionally, we provided practical insights of the described waste collection method in contrast with the state of practice (door-to-door collection).

The computational experiments in Section 6.2 confirm the advantages of the road-network representation such that the associated formulation (RN) outperforms its customer-based counterpart (CG) and provides a more intuitive characterization of the actual network. In brief, RN was able to find more often optimal and feasible solutions, obtained lower average and worst gaps (reported by Gurobi) and was faster in proving optimality. Furthermore, the quality of the solutions found by RN was better or equally good for all instances.

The proposed heuristic method (H) provides good solutions for the Cm-CTP-R and can handle the large instances that RN fails to solve to optimality or even fails to find a feasible solution within the given time limit. This method was able to find optimal solutions for 13 out of the 20 instances solved to optimality by RN and reported optimality gaps below 1.7% for the remaining instances. In addition, it found better solutions for most of the instances for which the exact method failed at proving optimality within the given time limit. From a practical perspective, we observe that the larger the maximum walking distance γ, the higher the savings in collection time with respect to door-to-door collection ($\gamma = 0$).

The presented MILP formulation and heuristic method could be further extended to accommodate other waste collection methods. For instance, an interesting concept results from introducing intermediate disposal facilities (Markov et al., 2016; Ramos et al., 2020) and an heterogeneous fleet of vehicles consisting of small vehicles, potentially electric, that bring the waste to the intermediate facilities and large ones that empty them and bring the waste to the disposal facility. To better represent reality, uncertain waste productions and travel times could be modeled by a set of discrete scenarios with the goal to minimize the worst or average cost over all of them. An interesting perspective to be integrated in the introduced problem is the maximization of residents' satisfaction via a multi-objective approach.

Acknowledgments

The authors gratefully acknowledge the support of Innosuisse, Switzerland under grant 36157.1 IP-EE. They also appreciate 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.

Álvarez-Miranda, E., & Sinnl, M. (2019). A note on computational aspects of the Steiner traveling salesman problem. *International Transactions in Operational Research*, *26*(4), 1396–1401.

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (2003). Implementing the Dantzig-Fulkerson-Johnson algorithm for large traveling salesman problems. *Mathematical Programming*, *97*(1), 91–153.

Archetti, C., Savelsbergh, M. W., & Speranza, M. G. (2006). Worst-case analysis for split delivery vehicle routing problems. *Transportation Science*, *40*(2), 226–234.

Baldacci, R., Boschetti, M. A., Maniezzo, V., & Zamboni, M. (2005). Scatter search methods for the covering tour problem. In *Metaheuristic optimization via memory and evolution* (pp. 59–91). Springer.

Bang-Jensen, J., & Gutin, G. Z. (2008). *Digraphs: Theory, algorithms and applications*. Springer.

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.

Ben Ticha, H., Absi, N., Feillet, D., Quilliot, A., & Van Woensel, T. (2019). A branch-and-price algorithm for the vehicle routing problem with time windows on a road network. *Networks*, *73*(4), 401–417.

Chen, P., Golden, B., Wang, X., & Wasil, E. (2017). A novel approach to solve the split delivery vehicle routing problem. *International Transactions in Operational Research*, *24*(1–2), 27–41.

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

Cornuéjols, G., Fonlupt, J., & Naddef, D. (1985). The traveling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming*, *33*(1), 1–27.

Cubillos, M., & Wøhlk, S. (2020). Solution of the maximal covering tour problem for locating recycling drop-off stations. *Journal of the Operational Research Society*, 1–16.

Current, J. R. (1981). *Multiobjective design of transportation networks* (Ph.D. thesis), Department of Geography and Environmental Engineering, Johns Hopkins University.

Current, J. R., & Schilling, D. A. (1989). The covering salesman problem. *Transportation Science*, *23*(3), 208–213.

Current, J. R., & Schilling, D. A. (1994). The median tour and maximal covering tour problems: Formulations and heuristics. *European Journal of Operational Research*, *73*(1), 114–126.

Davoodi, S. M. R., & Goli, A. (2019). An integrated disaster relief model based on covering tour using hybrid Benders decomposition and variable neighborhood search: Application in the Iranian context. *Computers & Industrial Engineering*, *130*, 370–380.

Dell'Amico, M., Maffioli, F., & Värbrand, P. (1995). On prize-collecting tours and the asymmetric travelling salesman problem. *International Transactions in Operational Research*, *2*(3), 297–308.

Fleischmann, B. (1985). A cutting plane procedure for the travelling salesman problem on road networks. *European Journal of Operational Research*, *21*(3), 307–317.

Gendreau, M., Laporte, G., & Semet, F. (1997). The covering tour problem. *Operations Research*, *45*(4), 568–576.

Ghiani, G., Laganà, D., Manni, E., & Triki, C. (2012). Capacitated location of collection sites in an urban waste management system. *Waste Management*, *32*(7), 1291–1296.

Glize, E., Roberti, R., Jozefowiez, N., & Ngueveu, S. U. (2020). Exact methods for mono-objective and bi-objective multi-vehicle covering tour problems. *European Journal of Operational Research*, *283*(3), 812–824.

Ha, M. H., Bostel, N., Langevin, A., & Rousseau, L. M. (2013). An exact algorithm and a metaheuristic for the multi-vehicle covering tour problem with a constraint on the number of vertices. *European Journal of Operational Research*, *226*(2), 211–220.

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

Huang, Y., Zhao, L., Van Woensel, T., & Gross, J. P. (2017). Time-dependent vehicle routing problem with path flexibility. *Transportation Research, Part B (Methodological)*, *95*, 169–195.

Jozefowiez, N. (2014). A branch-and-price algorithm for the multivehicle covering tour problem. *Networks*, *64*(3), 160–168.

Kammoun, M., Derbel, H., Ratli, M., & Jarboui, B. (2017). An integration of mixed VND and VNS: The case of the multivehicle covering tour problem. *International Transactions in Operational Research*, *24*(3), 663–679.

Karaoğlan, İ., Erdoğan, G., & Koç, Ç. (2018). The multi-vehicle probabilistic covering tour problem. *European Journal of Operational Research*, *271*(1), 278–287.

Labbé, M., & Laporte, G. (1986). Maximizing user convenience and postal service efficiency in post box location. *Belgian Journal of Operations Research, Statistics, and Computer Science*, *26*(2), 21–36.

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

Letchford, A. N., Nasiri, S. D., & Theis, D. O. (2013). Compact formulations of the Steiner traveling salesman problem and related problems. *European Journal of Operational Research*, *228*(1), 83–92.

Margolis, J. T., Song, Y., & Mason, S. J. (2022). A multi-vehicle covering tour problem with speed optimization. *Networks*, *79*(2), 119–142.

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.

Nagy, G., & Salhi, S. (2007). Location-routing: Issues, models and methods. *European Journal of Operational Research*, *177*(2), 649–672.

Naji-Azimi, Z., Renaud, J., Ruiz, A., & Salari, M. (2012). A covering tour approach to the location of satellite distribution centers to supply humanitarian aid. *European Journal of Operational Research*, *222*(3), 596–605.

Oliveira, W. A. d., Moretti, A. C., & Reis, E. F. (2015). Multi-vehicle covering tour problem: Building routes for urban patrolling. *Pesquisa Operacional*, *35*(3), 617–644.

OpenStreetMap (2022). https://www.openstreetmap.org.

Orloff, C. (1974). A fundamental problem in vehicle routing. *Networks*, *4*(1), 35–64.

Pham, T. A., Hà, M. H., & Nguyen, X. H. (2017). Solving the multi-vehicle multi-covering tour problem. *Computers & Operations Research*, *88*, 258–278.

Prodhon, C., & Prins, C. (2014). A survey of recent research on location-routing problems. *European Journal of Operational Research*, *238*(1), 1–17.

Ramos, T. R. P., Gomes, M. I., & Barbosa-Póvoa, A. P. (2020). A new matheuristic approach for the multi-depot vehicle routing problem with inter-depot routes. *OR Spectrum*, *42*(1), 75–110.

Rodrigues, S., Martinho, G., & Pires, A. (2016). Waste collection systems. Part A: A taxonomy. *Journal of Cleaner Production*, *113*, 374–387.

Salhi, S., & Nagy, G. (1999). Consistency and robustness in location-routing. *Studies in Locational Analysis*, (13), 3–19.

Schittekat, P., Kinable, J., Sörensen, K., Sevaux, M., Spieksma, F., & Springael, J. (2013). A metaheuristic for the school bus routing problem with bus stop selection. *European Journal of Operational Research*, *229*(2), 518–528.

Schneider, M., & Drexl, M. (2017). A survey of the standard location-routing problem. *Annals of Operations Research*, *259*(1), 389–414.

Tavares, G., Zsigraiova, Z., Semiao, V., & Carvalho, M. d. G. (2009). Optimisation of MSW collection routes for minimum fuel consumption using 3D GIS modelling. *Waste Management*, *29*(3), 1176–1185.

Tralhão, L., Coutinho-Rodrigues, J., & Alçada-Almeida, L. (2010). A multiobjective modeling approach to locate multi-compartment containers for urban-sorted waste. *Waste Management*, *30*(12), 2418–2429.

Vidal, T. (2022). Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. *Computers & Operations Research*, *140*, Article 105643.

Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., & Rei, W. (2012). A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, *60*(3), 611–624.