Published in Transportation research part B: methodological, 2016, vol. 85, pp. 256-273 which should be cited to refer to this work

Integrating a heterogeneous fixed fleet and a flexible assignment of destination depots in the waste collection VRP with intermediate facilities

Iliya Markov[a,b,*], Sacha Varone[b], and Michel Bierlaire[a]

[a]Transport and Mobility Laboratory
School of Architecture, Civil and Environmental Engineering
École Polytechnique Fédérale de Lausanne
Station 18, 1015 Lausanne, Switzerland

[b]Haute École de Gestion de Genève
University of Applied Sciences Western Switzerland (HES-SO)
Campus Battelle, Bâtiment F, Route de Drize 7, 1227 Carouge, Switzerland

iliya.markov@epfl.ch, sacha.varone@hesge.ch, michel.bierlaire@epfl.ch

May 12, 2015

Abstract

We consider a complex recyclable waste collection problem that extends the class of vehicle routing problems with intermediate facilities by integrating a heterogeneous fixed fleet and a flexible assignment of destination depots. Several additional side constraints, such as a mandated break period contingent on tour start time, multiple vehicle capacities and site dependencies are also included. This specific problem was inspired by a real-world application and does not appear in the literature. It is modeled as an MILP which is enhanced with several valid inequalities. Due to the rich nature of the problem, state-of-the-art commercial solvers are only able to tackle instances of small to medium size. To solve realistic instances, we propose a local search heuristic capable of systematically treating all problem features and general enough to respond to the varying characteristics of the case study regions for which it is intended. The results show that the heuristic achieves optimality on small random instances, exhibits competitive performance in comparison to state-of-the-art solution methods for special cases of our problem, and leads to important savings in the state of practice. Moreover, it highlights and quantifies the savings from allowing a flexible assignment of destination depots. The data from the state of practice comes from a recyclable waste collection company in Geneva, Switzerland.

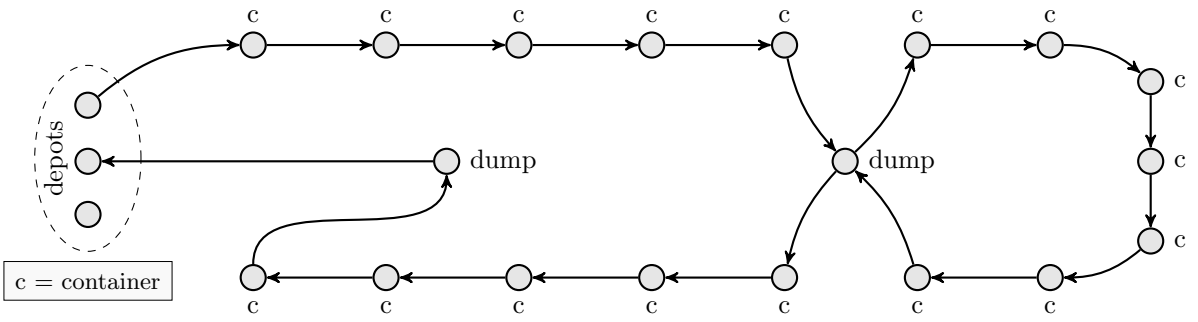
Keywords: vehicle routing; intermediate facilities; heterogeneous fixed fleet; flexible assignment of destination depots; local search; waste collection

1 Introduction

This article proposes a solution to a complex waste collection problem, in which recyclable waste is collected and transported using a heterogeneous fixed fleet of vehicles with different volume and weight

*Corresponding author: TRANSP-OR, ENAC, EPFL, GC B3 445, Station 18, 1015 Lausanne, Switzerland
Tel: +41 79 886 52 53; Email: iliya.markov@epfl.ch

Figure 1: Tour Example Illustration



capacities, fixed costs, unit-distance running costs and unit-time wage rates. As both waste containers and collection vehicles are flow-specific, the problem can be decomposed and solved separately for each waste flow.

As shown in Figure 1[in the final version ensure tables/figures/algorithms appear after they have been introduced in the text], each vehicle tour starts and ends at one of several depots, not necessarily the same, and is a sequence of collections followed by disposals at the available dumps. All collections are of the same waste flow and all visited dumps accept the latter. There is a mandatory visit to a dump just before the end of a tour, i.e. a tour terminates with an empty vehicle. Dumps are recycling plants. There could be multiple dumps for the collected waste flow and they can be used when and as needed during a tour. We consider time windows for the depots, collection points and dumps. A tour is limited only by the legal duration of the working day, which is roughly divided into two parts by a mandated break. Moreover, due to the specificities of the collection regions, we consider site dependencies. Mountainous terrain and narrow streets, for example, are inaccessible with big collector trucks.

Waste collection problems are often represented as VRPs with intermediate facilities (VRP-IF), with the latter representing the dumps where vehicles can stop to empty whenever needed during a tour. However, despite the wide practical application of these problems, many important realistic features are often omitted. For example, the literature on waste collection VRP assumes homogeneous fleets, while, on the other hand, heterogeneous fleet VRP problems omit other important features present in waste collection, such as intermediate facilities and complex temporal constraints. In addition, different collection regions often have peculiarities that make the direct application of existing solutions impractical.

The contribution of this work is the integration of a heterogeneous fixed fleet and a flexible assignment of destination depots in the vehicle routing problem with intermediate facilities. The former reflects the fact that, in industry, fleets either start as heterogeneous or become such as new vehicles are added or old ones are replaced, which introduces another level of complexity for the solution algorithms. The latter mimics situations that occur in our case study collection regions, some of which are dense city centers, while others cover large rural areas with long tours that do not necessarily terminate at the origin depot. According to the current state of practice, a driver who starts a tour from their "home" depot may occasionally have to terminate their tour and sleep over at another depot, from which they start on the next day. Given the relative inconvenience of such policies for the driver, we introduce a relocation term in the objective function, which incentivizes through a cost component, rather than enforcing, the vehicle to return to the origin depot. We thus keep the possibility of such policies, but minimize their occurrence.

We model the problem as a mixed binary linear program which is enhanced with several valid inequalities. Due to the rich nature of the problem, state-of-the-art commercial solvers are only able to tackle instances of small to medium size. For solving instances of realistic size, we propose a local search

heuristic[name the heuristic] which is capable of systematically treating all problem features and is general enough to be deployed to case studies with very different instance topologies. The heuristic achieves optimality on small random instances and exhibits competitive performance in comparison to state-of-the-art solution methods for special cases of our problem. It highlights and quantifies the benefits of the flexible assignment of destination depots, and leads to important financial savings in the current state of industry practice in the canton of Geneva, Switzerland, for which detailed data for comparison is available.

The remainder of this article is organized as follows. Section 2 is a brief analysis of the related literature. Sections 3 and 4 present our exact and heuristic approaches to solving the problem, respectively. Section 5 discusses the results from the numerical experiments, and Section 6 concludes and outlines directions for future research.

2 Related Literature

Although research on waste collection VRP spans several decades, the problem has been less studied than other VRP variants. One of the first applications is that of Beltrami and Bodin (1974) who solve a periodic VRP-IF (PVRP-IF) applied to a waste collection problem in New York. Bard et al. (1998a) and Bard et al. (1998b) consider a distribution context with replenishment facilities, in the latter case integrated in an inventory management framework. Angelelli and Speranza (2002b) apply a modification of Cordeau et al.'s (1997) unified tabu search (TS) algorithm to a PVRP-IF with features such as service durations and a maximum tour duration. In Angelelli and Speranza (2002a), this framework is used to analyze the operational cost benefits of different waste collection policies in Val Trompia, Italy and Antwerp, Belgium.

Kim et al. (2006) include time windows and a driver break in the waste collection VRP-IF, explicitly considering also features such as tour compactness and workload balancing. Their solution approach, an extension of Solomon's (1987) insertion algorithm followed by simulated annealing, leads to a significant reduction in the number of tours and substantial financial savings at a major US waste collection company (see Sahoo et al., 2005). Kim et al. (2006) are also the first to propose a set of 10 benchmark instances for the VRP-IF, involving up to 2092 stops and 19 intermediate disposal facilities. The multi-objective genetic algorithm of Ombuki-Berman et al. (2007), the variable neighborhood tabu search of Benjamin (2011) and the adaptive large neighborhood search (ALNS) of Buhrkal et al. (2012) are also applied on these instances, leading to distance improvements of 10-15% and using fewer vehicles. Buhrkal et al.'s (2012) approach also leads to a distance improvement of 30-45% at a Danish waste collection company.

Crevier et al. (2007) propose the multi-depot VRP with inter-depot routes (MDVRPI). Although the setup is closely related, it was originally applied in a distribution context. The MDVRPI is non-periodic, no time windows or driver breaks are considered and, in the general case, depots and intermediate facilities coincide. Crevier et al. (2007) use the adaptive memory (AM) principle of Rochat and Taillard (1995) and decompose the problem into multi-depot, single-depot and inter-depot subproblems which are solved using Cordeau et al.'s (1997) TS. A solution to the MDVRPI is obtained through a set covering formulation and improved by a modified version of the TS.

Crevier et al. (2007) create two sets of MDVRPI instances with 48 to 288 customers and a fixed homogeneous fleet stationed at one depot, with the rest of the depots acting only as intermediate facilities. These instances are used by Tarantilis et al. (2008) and Hemmelmayr et al. (2013) who propose, respectively, a hybrid guided local search and a variable neighborhood search (VNS) with a dynamic programming procedure for the insertion of the intermediate facilities in the tours. Both articles report improvements over the results of Crevier et al. (2007) with computation times close to one hour for the largest instances. Muter et al. (2014) develop a branch-and-price algorithm for the MDVRPI and solve several sets of instances derived from Crevier et al.'s (2007) benchmarks to

optimality. Only Hemmelmayr et al. (2013) apply their methodology to a PVRP-IF faced by a waste collection company and achieve a 25% reduction in the routing cost. Hemmelmayr et al. (2014) combine this problem with the bin allocation problem and study the cost trade-off between less frequent visits and larger bin sizes. They use a matheuristic with a VNS for the routing problem and a mathematical model for the bin allocation problem, and compare a hierarchical and an integrated approach.

Another related problem class is the routing of electric or alternative fuel vehicles, where we have recharging or refueling decisions in lieu of emptying decisions. Conrad and Figliozzi (2011) consider the recharging VRP (RVRP), where electric vehicles can recharge at customer locations with time windows. Erdoğan and Miller-Hooks (2012) consider the green VRP (G-VRP), where vehicles use a sparse alternative fuel infrastructure. Results on medium-size random instances show that spatial characteristics have significant impact on the optimality gap, which appears to be related to the number of facilities. Larger instances are used to analyze the effects of increasing the number of customers, facility availability and driving range limits.

Schneider et al. (2014a) solve the electric VRP with time windows and recharging stations (E-VRPTW). The problem features variable recharging times based on remaining battery charge and a hierarchical objective function minimizing number of vehicles first and travel distance second. The proposed hybrid VNS/TS improves the results of Erdoğan and Miller-Hooks (2012) by 8-15% and obtains competitive results on the MDVRPI sets of Crevier et al. (2007) and Tarantilis et al. (2008). Schneider et al. (2014b) combine recharging and reloading facilities in the VRP with intermediate stops (VRPIS). Contrary to the E-VRPTW, here the objective function is weighted rather than hierarchical. The authors propose an ALNS, which is able to match or improve the results of Schneider et al. (2014a) on the G-VRP instances at a fraction of the computation time. Convincing results are also obtained for the MDVRPI instances of Crevier et al. (2007) and Tarantilis et al. (2008).

Regarding the vehicle fleets, Kim et al. (2006) and the related papers on the VRP-IF assume an unlimited homogeneous fleet. The PVRP-IF of Angelelli and Speranza (2002b), the MDVRPI, RVRP, G-VRP, E-VRPTW and VRPIS also assume a homogeneous fleet, albeit limited. However, in industry vehicle fleets are rarely homogeneous. Few studies consider intermediate facilities in a more sophisticated context, notably Coene et al. (2010) and Prescott-Gagnon et al. (2014), both of which are case studies. The former treats a heterogeneous fixed fleet in a single-depot periodic VRP applied to a Belgian bio-waste collection company. The latter consider multiple depots but with no comparison against benchmark instances or the state of practice.

Taillard (1999) was the first to formally define the heterogeneous fixed fleet VRP (HFFVRP). Being a generalization of the vehicle fleet mix problem (VFMP), the HFFVRP is NP-hard and more difficult than the classical VRP or the VFMP. Taillard's (1999) solution approach relies on heuristic column generation with AM, and vehicle assignment costs are calculated at each iteration. He adapts the eight largest VFMP instances of Golden et al. (1984) to the HFFVRP by specifying the number of vehicles of each type and their variable costs. The best heuristic approaches on these benchmarks are due to Penna et al. (2013) and Subramanian et al. (2012), the latter also being the fastest. The only fully exact method is that of Baldacci and Mingozzi (2009). They prove the optimality of seven of the best known solutions to the instances with variable costs only, and six in the case where both fixed and variable costs are considered.

To our knowledge, Kek et al. (2008) is the only study to consider the flexible assignment of depots—a situation that appears in our problem, and would allow for vehicles to start and end at different depots, while being able to freely visit any dump for emptying. Kek et al.'s (2008) solution approach is based on a network model and a branch-and-bound algorithm, branching on the node with the best bound. On small randomly generated instances, the authors demonstrate that the flexible strategy outperforms a fixed one by almost 50%.

The originality of our problem is thus reinforced by the general lack of literature treating the heterogeneous fixed fleet VRP-IF despite its wide practical application. The flexible assignment of destination

depots is a characteristic that appears less frequently in practice and has therefore been largely ignored in the literature. Yet, the intelligent choice of destination depots can lead to important financial savings. Some of the waste collectors in our case study regions need such flexibility but are unable to assess its benefits. This article will therefore highlight and quantify the value of such strategies through a systematic solution approach.

3 Exact Approach

The formulation we propose introduces several extensions to the model of Sahoo et al. (2005), including multiple origins and destinations, multiple capacities, site dependencies, a maximum tour duration, a richer objective function capturing the costs faced by a realistic firm, and the elimination of the constraints calculating the necessary number of disposal trips for each vehicle. The break is modeled in a way similar to Buhrkal et al. (2012) but without imposing a hard time window. In what follows we present the model formulation as well as several problem-specific valid inequalities with a significant impact on computation time. [impact to be measured with new experiments]

3.1 Mathematical Formulation

Formally, we define the problem on a directed multigraph $G(N, A)$, with $N = O' \cup O'' \cup D \cup P$, where O' is the set of origin depots, O'' is the set of destination depots, D is the set of dumps, P is the set of containers, and $A = \{(i, j) \mid \forall i, j \in N\}$ is the set of arcs. Each dump in the set D is replicated as many times as the maximum number of dump visits by any vehicle k, with an upper bound given by ceiling of the ratio of total demand over the capacity of the smallest vehicle [this has to be thought out more carefully].

The arcs are associated with an asymmetric distance matrix Π, where π_{ij} is the length of arc (i, j). Each vehicle may have a different average speed, which results in a vehicle specific travel time matrix T_k, where τ_{ijk} is the travel time of vehicle k on arc (i, j). Each point has a single time window $[\lambda_i, \mu_i]$, where λ_i and μ_i stand for the earliest and latest possible start-of-service time. Start of service after μ_i is not allowed and if the vehicle arrives before λ_i it has to wait. Service duration for each point is denoted by ε_i, and the pickup volume and weight by ρ_i^v and ρ_i^w, respectively. The service duration for containers is mostly influenced by the type of container, e.g. underground or overground, and at dumps by factors such as weighting and billing, hence it is only indexed over the set N. Service duration at depots is zero.

There is a heterogeneous fixed fleet K, with each vehicle defined by its capacity in terms of maximum volume Ω_k^v and weight Ω_k^w, a fixed deployment cost ϕ_k, a unit-distance running cost β_k, and a unit-time wage rate θ_k. We assume the driver-to-vehicle assignment as given and thus associate, for an instance that is being solved, the wage rate directly to the vehicle. There is a maximum tour duration of H, and a break of duration δ must be taken after η hours of continuous work, which divides the working day into two roughly equal halves. Site dependencies are described by a binary flag α_{ijk} whose value is 1 if arc (i, j) is accessible for vehicle k, and 0 otherwise.

We introduce the following binary decision variables: $x_{ijk} = 1$ if vehicle k traverses arc (i, j), 0 otherwise; $z_{ijk} = 1$ if i and j are, respectively, the origin and destination of vehicle k, 0 otherwise; $b_{ijk} = 1$ if vehicle k takes a break on arc (i, j), 0 otherwise; $y_k = 1$ if vehicle k is used, 0 otherwise. Three groups of continuous variables, Q_{ik}^v, Q_{ik}^w and S_{ik}, are defined to track the cumulative volume and weight and the start-of-service time at point i for vehicle k. Table 1 is a summary of the used notations.

The objective function (1) minimizes two terms. The first one is the sum of fixed cost, unit-distance running cost and unit time wage rates for all used vehicles. The second one is the total cost of relocation multiplied by the weight factor Ψ. Obviously, if i and j in the second term do not refer to the same

Table 1: Mathematical Model Notations

Sets			
O'	set of origins	O''	set of destinations
D	set of dumps	P	set of containers
N	$= O' \cup O'' \cup D \cup P$	K	set of vehicles
Parameters			
π_{ij}	length of arc (i, j)		
α_{ijk}	$= 1$ if arc (i, j) is accessible for vehicle k, 0 otherwise		
τ_{ijk}	travel time of vehicle k on arc (i, j)		
ε_i	service duration at point i		
$[\lambda_i, \mu_i]$	lower and upper time window bound at point i		
H	maximum tour duration		
η	maximum continuous work limit after which a break is due		
δ	break duration		
ρ_i^v, ρ_i^w	pickup volume and weight at point i		
Ω_k^v, Ω_k^w	volume and weight capacity of vehicle k		
ϕ_k	fixed cost of vehicle k		
β_k	unit-distance running cost of vehicle k		
θ_k	unit-time wage rate of vehicle k		
Ψ	weight of relocation cost term		
Decision Variables			
x_{ijk}	$= \begin{cases} 1 & \text{if vehicle } k \text{ traverses arc } (i,j) \\ 0 & \text{otherwise} \end{cases}$		
z_{ijk}	$= \begin{cases} 1 & \text{if } i \text{ and } j \text{ are, respectively, the origin and destination of vehicle } k \\ 0 & \text{otherwise} \end{cases}$		
b_{ijk}	$= \begin{cases} 1 & \text{if vehicle } k \text{ takes a break on arc } (i,j) \\ 0 & \text{otherwise} \end{cases}$		
y_k	$= \begin{cases} 1 & \text{if vehicle } k \text{ is used} \\ 0 & \text{otherwise} \end{cases}$		
Q_{ik}^v	cumulative volume on vehicle k at point i (continuous)		
Q_{ik}^w	cumulative weight on vehicle k at point i (continuous)		
S_{ik}	start-of-service time of vehicle k at point i (continuous)		

physical depot, there will be a positive distance and travel time between them, which is thus explicitly integrated into the total cost. This particular formulation reflects fact that our case study includes a wide service area with several waste collection companies and a mix of urban and rural regions and it is not always optimal, especially in rural areas, for a vehicle to return to its origin depot. If the driver terminates their tour at a depot different from their "home" depot, they can sleep over and start a new tour from there on the next day. The relocation cost term is a disincentive to such practices, and ensures that they will occur only if there is a sufficient financial benefit. The weight factor Ψ can be determined by the company.

$$\begin{aligned} \text{Min} \quad f = \sum_{k \in K} \left(\phi_k y_k + \beta_k \sum_{i \in N} \sum_{j \in N} \pi_{ij} x_{ijk} + \theta_k \left(\sum_{j \in O''} S_{jk} - \sum_{i \in O'} S_{ik} \right) \right) \\ + \Psi \sum_{k \in K} \sum_{i \in O'} \sum_{j \in O''} \left(\beta_k \pi_{ji} + \theta_k \tau_{jik} \right) z_{ijk} \end{aligned} \tag{1}$$

The constraints can be split into several categories with the first category consisting of basic vehicle routing constraints. Equalities (2) impose that each point should be served by exactly one vehicle. Equalities (3) and (4) ensure that if a vehicle is used, its tour starts at an origin and ends at a

destination with a visit to a dump immediately before that[THERE SHOULD BE SOME FIXING HERE WITH RESPECT TO WHERE THE VEHICLE IS AND IF IT SHOULD BE FORCED TO RETURN TO A SPECIFIC DEPOT]. Constraints (5) and (6) forbid returning to an origin or leaving a destination. Flow conservation is represented by constraints (7). The link between the variables x_{ijk} and z_{ijk} is achieved through constraints (8).

$$\sum_{k\in K}\sum_{j\in D\cup P} x_{ijk} = 1, \qquad \forall i \in P \tag{2}$$

$$\sum_{i\in O'}\sum_{j\in P} x_{ijk} = y_k, \qquad \forall k \in K \tag{3}$$

$$\sum_{i\in D}\sum_{j\in O''} x_{ijk} = y_k, \qquad \forall k \in K \tag{4}$$

$$\sum_{i\in N} x_{ijk} = 0, \qquad \forall k \in K, j \in O' \tag{5}$$

$$\sum_{j\in N} x_{ijk} = 0, \qquad \forall k \in K, i \in O'' \tag{6}$$

$$\sum_{i\in N\setminus O''} x_{ijk} = \sum_{i\in N\setminus O'} x_{jik}, \qquad \forall k \in K, j \in D\cup P \tag{7}$$

$$\sum_{m\in P} x_{imk} + \sum_{m\in D} x_{mjk} - 1 \leqslant z_{ijk}, \qquad \forall k \in K, i \in O', j \in O'' \tag{8}$$

Site dependencies are enforced by constraints (9).

$$x_{ijk} \leqslant \alpha_{ijk}, \qquad \forall k \in K, i \in N\setminus O'', j \in N\setminus O' \tag{9}$$

In the context of vehicle capacities, inequalities (10) and (11) limit, respectively, the cumulative volume and weight on the vehicle at each point, while equalities (12) and (13) reset them to zero at the dumps, origins and destinations. Keeping track of the cumulative volume and weight on the vehicle is achieved by constraints (14) and (15).

$$\rho_i^v \leqslant Q_{ik}^v \leqslant \Omega_k^v, \qquad \forall k \in K, i \in P \tag{10}$$

$$\rho_i^w \leqslant Q_{ik}^w \leqslant \Omega_k^w, \qquad \forall k \in K, i \in P \tag{11}$$

$$Q_{ik}^v = 0, \qquad \forall k \in K, i \in N\setminus P \tag{12}$$

$$Q_{ik}^w = 0, \qquad \forall k \in K, i \in N\setminus P \tag{13}$$

$$Q_{ik}^v + \rho_j^v \leqslant Q_{jk}^v + \Omega_k^v\left(1 - x_{ijk}\right), \qquad \forall k \in K, i \in N\setminus O'', j \in P \tag{14}$$

$$Q_{ik}^w + \rho_j^w \leqslant Q_{jk}^w + \Omega_k^w\left(1 - x_{ijk}\right), \qquad \forall k \in K, i \in N\setminus O'', j \in P \tag{15}$$

The next four constraints express the temporal characteristics of the problem. Inequalities (16) calculate the start-of-service time at each point, including service duration and a possible break duration. In addition, these constraints eliminate the possibility of subtours and ensure that a point will not be visited more than once by the same vehicle. Constraints (17), (18) and (19) enforce the time windows and maximum tour duration. We assume that the lower time window bound is restrictive at the origins and the upper one at the destinations.

$$S_{ik} + \varepsilon_i + \delta b_{ijk} + \tau_{ijk} \leqslant S_{jk} + \left(\max_{m\in O''}\mu_m\right)\left(1 - x_{ijk}\right), \qquad \forall k \in K, i \in N\setminus O'', j \in N\setminus O' \tag{16}$$

$$\lambda_i \sum_{j\in N\setminus O'} x_{ijk} \leqslant S_{ik}, \qquad \forall k \in K, i \in N\setminus O'' \tag{17}$$

$$S_{jk} \leqslant \mu_j \sum_{i\in N\setminus O''} x_{ijk}, \qquad \forall k \in K, j \in N\setminus O' \tag{18}$$

$$\sum_{j \in O''} S_{jk} - \sum_{i \in O'} S_{ik} \leqslant \mathrm{H}, \qquad \forall k \in K \tag{19}$$

The next block of constraints determines the arc on which a break is due. Breaks are modeled on the arcs as in much of the vehicle routing literature and can in practice be taken on the arcs' tails. Constraints (20) and (21) limit the arcs on which the break can be taken so as it is taken as late as possible. Inequalities (22) impose that the vehicle can only take a break on the arcs it traverses. Finally, inequalities (23) ensure that the break is actually taken if the vehicle tour is longer than the maximum continuous work limit η. The motivation to model the break as late as possible is two-fold. First, few containers have very restrictive time windows (e.g. near schools or during market days) and there would always be other containers in the area that can be served with little or no waiting time. Therefore, it can be safely assumed that waiting time will never be sufficient to accommodate a break, which is typically an hour long. Secondly, this approach allows the working day to be divided into two periods of an approximate length η, thus avoiding the requirement for a second long break [big M and tau related].

$$\left(S_{ik} - \sum_{m \in O'} S_{mk}\right) + \varepsilon_i - \eta \leqslant (\mathrm{H} - \eta - \xi)\,(1 - b_{ijk})\,, \qquad \forall k \in K, i \in N \setminus O'', j \in N \setminus O' \tag{20}$$

$$\eta - \left(S_{jk} - \sum_{m \in O'} S_{mk}\right) \leqslant \left(\max_{m \in O'} \mu_m + \eta\right)(1 - b_{ijk})\,, \qquad \forall k \in K, i \in N \setminus O'', j \in N \setminus O' \tag{21}$$

$$b_{ijk} \leqslant x_{ijk}, \qquad \forall k \in K, i, j \in N \tag{22}$$

$$\left(\sum_{j \in O''} S_{jk} - \sum_{i \in O'} S_{ik}\right) - \eta \leqslant (\mathrm{H} - \eta) \sum_{i \in N \setminus O''} \sum_{j \in N \setminus O'} b_{ijk}, \quad \forall k \in K \tag{23}$$

In constraints (20, the value of ξ can be set to the fastest travel time on accessible arcs from i to a destination that passes through a dump. Finally, (24) to (26) establish the variable domains.

$$x_{ijk}, y_k, b_{ijk} \in \{0, 1\}, \qquad \forall k \in K, i, j \in N \tag{24}$$

$$z_{ijk} \in \{0, 1\}, \qquad \forall k \in K, i \in O', j \in O'' \tag{25}$$

$$Q^v_{ik}, Q^w_{ik}, S_{ik} \geqslant 0, \qquad \forall k \in K, i \in N \tag{26}$$

3.2 Variable Fixing and Valid Inequalities

We can exploit the special structure of our problem by fixing some of the binary variables and defining several valid inequalities that restrict the search space of some of the binary and continuous variables without eliminating any feasible solutions. We first set to zero binary variables linked to impossible traversals. Constraints (27) eliminate the possibility of loops. In a similar fashion, constraints (28), (29) and (30) forbid traveling from an origin to a dump or destination, from a container to a destination, and from a dump to another dump, respectively.

$$x_{iik} = 0, \qquad \forall k \in K, i \in N \tag{27}$$

$$x_{ijk} = 0, \qquad \forall k \in K, i \in O', j \in D \cup O'' \tag{28}$$

$$x_{ijk} = 0, \qquad \forall k \in K, i \in P, j \in O'' \tag{29}$$

$$x_{ijk} = 0, \qquad \forall k \in K, i \in D, j \in D \colon i \neq j \tag{30}$$

The presence of time windows allows us to fix time-window infeasible traversals. Constraints (31) express the fact that if by visiting point i as early as possible vehicle k cannot visit point j within its time window, then points i and j cannot be visited by the same vehicle k, i.e. arc (i, j) is not traversed

by vehicle k. These first two sets of rules can also be used to eliminate all the big M constraints (14, 15, 16, 20, 21, 23) for such variables as they become trivial.

$$x_{ijk} = 0, \qquad \forall k \in K, i \in N \setminus O'', j \in N \setminus O' \colon \lambda_i + \varepsilon_i + \tau_{ijk} > \mu_j \tag{31}$$

The first set of valid inequalities is used to restrict the start-of-service time search space. Inequalities (32) impose a lower bound, short of waiting times, on the difference between the start-of-service time at the origin and destination for each used vehicle. Then inequalities (33) and (34) calculate the latest possible start and earliest possible finish of each tour.[33 and 34, bigM and tau related]

$$\sum_{j \in O''} S_{jk} - \sum_{i \in O'} S_{ik} \geqslant \sum_{i \in N \setminus O''} \sum_{j \in N \setminus O'} x_{ijk}(\varepsilon_i + \tau_{ijk}), \qquad \forall k \in K \tag{32}$$

$$S_{ik} \leqslant \max_{m \in P^\star} (\mu_m - \tau_{imk})\, y_k, \qquad \forall k \in K, i \in O' \tag{33}$$

$$S_{jk} \geqslant \min_{m \in D^\star} (\lambda_m + \varepsilon_m + \tau_{mjk}) \sum_{m \in D^\star} x_{mjk}, \qquad \forall k \in K, j \in O'' \tag{34}$$

In constraints (33) $P^* \subset P$, where $\alpha_{imk} = 1$, and in constraints (33) $D^* \subset D$, where $\alpha_{mjk} = 1$. If the problem involves subsets of identical vehicles, the presence of symmetry can substantially reduce the effectiveness of the model. Let $K' \subset K$ represent a subset of identical vehicles and let $k'_g \in K'$, where $g \in 1, \ldots, |K'|$ introduces a simple ordering of the elements of K'. Then for each subset K' we apply constraints (35) or (36). These symmetry-breaking constraints specify that the first vehicle in K' executes the tour with the highest waste volume (weight), the second vehicle executes the tour with the second highest waste volume (weight), etc.

$$\sum_{i \in P} \sum_{j \in P \cup D} \rho_i^v x_{ijk'_g} \geqslant \sum_{i \in P} \sum_{j \in P \cup D} \rho_i^v x_{ijk'_{g+1}}, \qquad \forall g \in 1, \ldots, (|K'| - 1) \tag{35}$$

$$\sum_{i \in P} \sum_{j \in P \cup D} \rho_i^w x_{ijk'_g} \geqslant \sum_{i \in P} \sum_{j \in P \cup D} \rho_i^w x_{ijk'_{g+1}}, \qquad \forall g \in 1, \ldots, (|K'| - 1) \tag{36}$$

Symmetries will also result from the fact that dumps are replicated. In a similar way, we define $D' \subset D$ as a subset of replications of the same physical dump and let $i'_g \in D'$, where $g \in 1, \ldots, |D'|$ introduces a simple ordering of the elements of D'. Then for each subset D' we apply the lexicographical ordering constraints (37), which stipulate that a dump with a higher index should be preceded by a container with a higher index.

$$\sum_{j \in P} j x_{ji'_g k} \leqslant \sum_{j \in P} j x_{ji'_{g+1} k}, \qquad \forall k \in K, g \in 1, \ldots, (|D'| - 1) \tag{37}$$

The last set of valid inequalities concerns the dump visits. With (38) we impose that a dump may be visited at most once by a vehicle. With (39), on the other hand, we set for every vehicle the maximum number of trips from dumps to containers, which is one less the total number of dumps. A dump visit is thus reserved for the final trip in each tour.

$$\sum_{i \in P} x_{ijk} \leqslant 1, \qquad \forall k \in K, j \in D \tag{38}$$

$$\sum_{i \in D} \sum_{j \in P} x_{ijk} \leqslant |D| - 1, \qquad \forall k \in K \tag{39}$$

With the addition of the valid inequalities, a state-of-the-art commercial solver like Gurobi can handle instances with 10-15 containers, a depot, 4-5 dumps and 5 vehicles with the only critical resource being computation time. Computation times are influenced both by the instance sizes and by their spatial and temporal characteristics. We return to this question in Section 5.

4 Local Search Heuristic

The vehicle routing problem is well known to be NP-hard (see e.g. Garey and Johnson, 1979). Being a generalization thereof, our waste collection problem is even harder to solve. Moreover, realistic instances involving 50 or more containers and several depots, dumps and vehicles will translate into thousands of binary variables and tens of thousands of constraints. Therefore, for such cases, we develop a local search heuristic capable of systematically treating all problem features. It starts off by constructing an initial solution and then applies an iterative improvement procedure accepting infeasible intermediate solutions. The heuristic contains components that specifically tackle each feature of our problem and has been designed to be as independent as possible from the underlying instance topologies in order to be applicable to the wide array of geographies present in the waste collection regions of interest. It relies on conventional search operators and techniques. Therefore, the originality of the approach resides in their combined use to achieve efficiency on a new problem.

4.1 Feasibility

A solution to our problem is a set of tours. It is considered feasible if all tours that comprise it satisfy four criteria. First, start-of-service times should respect time windows. Secondly, tour duration should be shorter than or equal to the maximum tour duration. These two criteria may be thought of as expressing temporal feasibility. Thirdly, the volume and weight capacities of the vehicles may not be violated at any point. This can be ensured by inserting appropriate visits to the available dumps. We attach to this last criterion the condition that a tour should start from a specific depot, finish at one of the available [or a specific] depots and visit a dump before the end. Finally, site dependencies should be respected.

Every insertion or removal of a point from a tour, and every application of a neighborhood operator requires the recalculation of start-of-service and waiting times for all or part of the points in the tour. As shown in Algorithm 1, we consider a tour served by vehicle $k \in K$, for brevity tour k, represented as an ordered sequence of points $1, 2, \ldots, n-1, n$ indexed by i. The calculation begins by setting the start-of-service time at the origin, S_{1k}, as early as possible. For each subsequent point i, S_{ik} is tentatively calculated as the sum of the start-of-service time at point $i-1$, the service duration at point $i-1$, and the travel time from $i-1$ to i, i.e. $S_{ik} = S_{(i-1)k} + \varepsilon_{i-1} + \tau_{(i-1)ik}$. If the maximum continuous working time limit η expires between the start-of-service time at $i-1$ and the end-of-service time at i, in other words if $S_{(i-1)k} < S_{1k} + \eta$ and $S_{ik} + \varepsilon_i > S_{1k} + \eta$, we need to insert the required break before serving point i, which is achieved by incrementing S_{ik} by the break duration δ. Finally, if S_{ik} violates the lower time window bound λ_i, i.e. if $S_{ik} < \lambda_i$, we introduce waiting time w_{ik} at point i, equal to the difference $\lambda_i - S_{ik}$, and update S_{ik} to λ_i. Once all S_{ik} have been determined, we check if upper time window bounds μ_i are respected for all i. If this is the case, we apply forward time slack reduction on the tour, otherwise we declare the tour time-window infeasible.

Forward time slack, as described by Savelsbergh (1992), keeps track of the maximum amount each start-of-service time can be delayed without violating time windows on the tour. We will examine points sequentially in reverse order. If there is waiting at point i, there could be a non-zero slack at point $i-1$, because pushing $S_{(i-1)k}$ forward may eliminate or reduce waiting at i. We can push $S_{(i-1)k}$ forward by the amount of waiting at i, or until we reach the upper time window bound at $i-1$. The last operation is expressed as $S_{(i-1)k} = \min(S_{(i-1)k} + w_{ik}, \mu_{i-1})$, and it entails an update of $w_{(i-1)k}$ and w_{ik} to factor in the potential increase of waiting at $i-1$ and decrease of waiting at i. Let $S'_{(i-1)k}$ denote the original start-of-service time at point $i-1$ before slack reduction. Then, waiting at $i-1$ will be increased by the difference between $S_{(i-1)k}$ and $S'_{(i-1)k}$, and waiting at i will be reduced by the same difference. Finally, we need to artificially put $w_{1k} = 0$. Forward time slack reduction preserves time-window feasibility. Therefore, after the procedure it only remains to check if the tour's duration

Algorithm 1 Temporal feasibility algorithm

Input tour k as a sequence of points $1, 2, \ldots, n-1, n$
Output start-of-service times, waiting times and temporal feasibility of tour k

$S_{1k} \coloneqq \lambda_1$
for $i = 2, 3, \ldots, n-1, n$ in tour k **do**
    $S_{ik} \coloneqq S_{(i-1)k} + \varepsilon_{i-1} + \tau_{(i-1)ik}$
    if $S_{(i-1)k} < S_{1k} + \eta$ **and** $S_{ik} + \varepsilon_i > S_{1k} + \eta$ **then**
        $S_{ik} \coloneqq S_{ik} + \delta$
    end if
    if $S_{ik} < \lambda_i$ **then**
        $w_{ik} \coloneqq \lambda_i - S_{ik}$
        $S_{ik} \coloneqq \lambda_i$
    else
        $w_{ik} \coloneqq 0$;
    end if
end for
if $S_{ik} \leqslant \mu_i, \forall i$ **then**
    for $i = n, n-1, \ldots, 3, 2$ in tour k **do**
        $S'_{(i-1)k} \coloneqq S_{(i-1)k}$
        $S_{(i-1)k} \coloneqq \min\left(S_{(i-1)k} + w_{ik}, \mu_{i-1}\right)$
        $w_{(i-1)k} \coloneqq w_{(i-1)k} + \left(S_{(i-1)k} - S'_{(i-1)k}\right)$
        $w_{ik} \coloneqq w_{ik} - \left(S_{(i-1)k} - S'_{(i-1)k}\right)$
    end for
    $w_{1k} \coloneqq 0$
    if $S_{nk} - S_{1k} \leqslant \mathrm{H}$ **then**
        tour k is temporally feasible
    else
        tour k is duration infeasible
    end if
else
    tour k is time-window infeasible
end if

is feasible. If it is the case, we accept the tour as temporally feasible, otherwise we declare it duration infeasible. [note on break]

Verifying capacity feasibility is much more straightforward. At each point of the tour, we calculate the cumulative volume and weight loads, Q^v_{ik} and Q^w_{ik}, on the vehicle, resetting both to zero if the point is a dump. If, for any point i, $Q^v_{ik} > \Omega^v_k$ or $Q^w_{ik} > \Omega^w_k$ or a dump is not visited immediately before the destination, we declare the tour capacity infeasible. The logic behind site dependency feasibility is trivial. Implementation-wise, we construct vehicle tours only using accessible points. Inaccessibilities may occur with the application of inter-tour operators and a simple count of the number of inaccessible points is updated with the application of an operator. The latter is much more efficient than inspecting all tour points when accessibility feasibility needs to be verified. [note on accessibility]

4.2 Initial Solution Construction

Tour construction is performed sequentially. Initially, all containers belong to the pool of unassigned containers P, and all vehicles to the pool of unassigned vehicles K. A seed tour is created by assigning the cheapest feasible sequence of origin, container, dump and destination to the cheapest available

vehicle. All assigned vehicles and containers are removed from their respective unassigned pools. Once a seed tour k has been created, it is expanded using a simple feasibility preserving greedy insertion heuristic. At each iteration, we insert container $i \in P$ at the position j in the tour that would yield the smallest cost increase. The point at position j, as well as all subsequent points, are shifted to the right.

If no more feasible container insertions are possible and if infeasibility would result from capacity violation, we insert a dump using the same logic, otherwise we terminate the tour. In addition, we require that the dump cannot be inserted as an immediate predecessor or successor of another dump on the tour or just after the origin depot. Finally, to avoid a meaningless increase in the objective function, we require that after a dump insertion there should be at least one feasible container insertion. If this condition does not hold, the last inserted dump is removed and the tour is terminated. Tour construction stops when the pool of unassigned containers is empty, or the pool of unassigned vehicles is empty, or infeasibilities prohibit further insertions. When each tour is constructed, it is individually improved using the single-tour operators described below.

For a solution to be feasible, it is assumed that the fleet is sufficient to service all containers, which is actually the case for all benchmark instances considered in the numerical experiments presented in Section 5. Nevertheless, in a real-life application where, for a given instance, the fleet is insufficient, the provided solution will still respect all constraints for the containers in it.

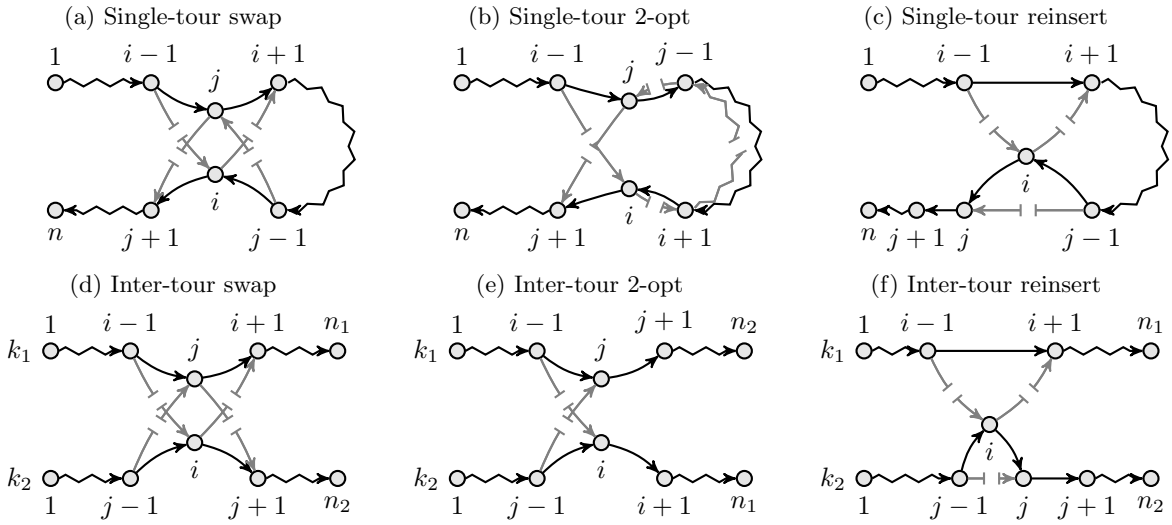
4.3 Iterative Solution Improvement

In order to keep the improvement phase as general as possible, we consider three neighborhoods—swap, 2-opt and reinsert, with each neighborhood using classical single- and inter-tour operators of the respective type. Figure 2 depicts the six operators with possible improvements from the application of each of them. The interrupted gray arcs form parts of the tours before the application of the operators. The resulting improved tours are given in solid black arcs. The application of an operator, whether single- or inter-tour, may lead to a feasible or an infeasible neighbor. If the neighbor solution is infeasible, its objective function (1) is multiplied by a factor *infFactor* larger than one. If the next neighbor is feasible, this factor is reduced by *infStepDown*, and if infeasible, it is increased by *infStepUp*. The factor *infFactor* will never drop below one.

To define the operators more precisely, the single-tour swap disconnects $i-1$ from i, i from $i+1$, $j-1$ from j, and j from $j+1$, and reconnects $i-1$ to j, j to $i+1$, $j-1$ to i, and i to $j+1$. Its inter-tour version works in exactly the same way with the only difference being that i and j belong to different tours. The single-tour 2-opt disconnects $i-1$ from i, and j from $j+1$, and reconnects $i-1$ to j, and i to $j+1$, thus reversing the orientation of the section $i, \ldots, j$, inclusive of i and j. The inter-tour 2-opt performs the same actions, where i and j belong to different tours, which results in the exchange of the end portions of the two affected tours, inclusive of i and j. Finally, the single-tour reinsert disconnects $i-1$ from i and i from $i+1$, reconnecting $i-1$ directly to $i+1$. Then it disconnects $j-1$ from j and reconnects $j-1$ to i and i to j. The logic of the inter-tour reinsert is the same, with i and j belonging to different tours. In essence, the last two operators remove a point i from its original position and insert it in the position of another point j, from the same or a different tour, pushing j to the right.

As described in Algorithm 2, the succession of neighborhoods (swap, 2-opt, reinsert in that order) is applied until either *maxIter* iterations or *maxNonImpIter* non-improving iterations has been reached. Each individual neighborhood is applied for *maxNbIter* iterations or *maxNbNonImpIter* non-improving iterations from the last visited local minimum, and at each neighborhood change we start again from the best feasible solution found so far and reset *infFactor*. For each neighbor, a random sample of single- and inter-tour moves of the current neighborhood is evaluated and the cheapest one, be it feasible or infeasible, in the last case evaluated after multiplication by *infFactor*, is accepted as the new incumbent. To prevent cycling and encourage diversification towards less explored areas of the search

Figure 2: Neighborhood Operators



This figure depicts improvements that can result from the application of each operator, with the interrupted gray arcs replaced by the solid black arcs.

space, a solution with the same objective value is not admitted more than once for a given number of iterations, denoted by *cycleFreq*. These non-admissible solutions are held in a ban list. When a new incumbent is generated, the ban list is updated to include its cost and exclude the cost older than *cycleFreq*.

At every *recoverFreq* iterations, and if the available fleet is heterogeneous (i.e. at least one vehicle is different from the rest), we evaluate and perform vehicle reassignments to tours. The vehicle reassignment evaluation procedure unassigns all assigned vehicles and starts inspecting the tours in a descending order of total load. For each consecutively examined tour, the best vehicle is assigned so that the assignment is feasible. The assignment feasibility is verified after applying the capacity recovery procedure described next. If no feasible assignment is possible, the available vehicle with the largest capacity is assigned to the tour. After the assignment, the capacity recovery procedure is rerun and the tour is individually improved. Alternatively, if the available fleet is homogeneous, we proceed directly to the capacity recovery procedure, followed by individual improvement of all tours.

The logic of the vehicle reassignment evaluation procedure is somewhat different compared to what would befit Taillard's (1999) HFFVRP formulation. Here the procedure tries to balance between two conflicting goals. Assuming a fleet with correlated characteristics, assigning cheaper vehicles to tours is counterbalanced by the necessity for more frequent visits to the dumps, because cheaper vehicles have smaller capacities. This is compounded by the fact that in a realistic scenario dumps (intermediate facilities) are located outside the collection area (for example in suburbs or industrial zones) instead of centrally as in the benchmark instances we see. Therefore, the logic of this procedure is different, as is its direct applicability to a pure HFFVRP formulation where reassignments have to be examined with every move. In our case, capacity infeasibility resulting after a move can easily be recovered by adding more dump visits or simply reordering them. Moreover, if a tour is attracting points from other tours, reassigning vehicles too often may have an adverse effect.

The capacity recovery procedure first removes all dump visits from the tour, after which it inspects the best dump insertion positions close to where capacity feasibility would become violated. This action serves two purposes. First, it removes unnecessary dump visits from short tours. Secondly, it may need to insert additional dump visits or reorder the dump visits in tours that may have been rendered

Algorithm 2 Iterative improvement procedure

Define: K is the set of all available vehicles
Input set of constructed tours $K' \in K$
Output set of improved tours $K'' \in K$

initialize *infFactor*
initialize ban list
initialize start neighborhood
currentIncumbent := solution from tour construction
for *maxIter* **do**
    for each neighborhood **do**
        for *maxNbIter* **do**
            N := random neighbor sample of *currentIncumbent*
            currentIncumbent := $\min(n)\{\text{cost}(n) \mid \forall n \in N : \text{cost}(n) \notin \text{ban list}\}$
            update *infFactor*
            update ban list
            if reached *recoverFreq* **then**
                vehicle reassignment evaluation procedure with cap. recovery and depot reasg.
                improve tours individually
                update ban list
            end if
            if reached *maxNbNonImpIter* **then**
                change neighborhood
                reset *currentIncumbent* to best feasible solution found so far
                reset *infFactor*
                break
            end if
        end for
        change neighborhood
        reset *currentIncumbent* to best feasible solution found so far
        reset *infFactor*
    end for
    if reached *maxNonImpIter* **then**
        break
    end if
end for

capacity infeasible by the neighborhood operators [expand explanations].

This is followed by the reassignment evaluation of destination depots, where all possibilities are evaluated given the fact that the number of depots tends to be small. The subsequent individual improvement may be able to recover infeasibilities related to maximum tour duration or time window violations. Moreover, if unassigned containers remain and can be feasibly inserted, new insertions are attempted during individual tour improvement before switching back from reinsert to swap. There is a penalty associated with unassigned containers, which encourages assignment with a near-guarantee of cost improvement. In the end, the logic of the local search heuristic is such that it remains fairly general rather than being tailored to a narrowly specified problem.

Table 2: Heuristic Parameter Values

Parameter	Value	Parameter	Value
maxNbIter	30	*infStepUp*	0.05
maxNbNonImpIter	7	*infStepDown*	0.02
maxIter	100 (10^a)	*cycleFreq*	∞
maxNonImpIter	15 (3^a)	*recoverFreq*	5
infFactor	1.10	sample size	10^b

[a] value for individual tour improvement
[b] at a given iteration, the chosen operator is evaluated on each node i for a random sample of 10 j nodes, see Figure 2

5 Numerical Experiments

To assess the quality of the heuristic, we compare its results to the optimal ones produced by the mathematical model on modifications of the small instances of Schneider et al. (2014a), to the best known solutions (BKS) of Kim et al.'s (2006) waste collection VRP-IF and Crevier et al.'s (2007) MDVRPI instances, and to the state of practice of a recyclable waste collector in the canton of Geneva, Switzerland. All tests were carried out on a 3.20 GHz Intel Core i5 machine with 8 GB of memory running a 64-bit Windows 7. The local search heuristic was coded in Java and the mathematical model was solved using the Gurobi 6.0.0 MIP solver via its Java API. The solver was warm-started, for each instance, with the solution obtained by the heuristic. Our tests are conducted using the same values of the heuristic parameters presented in Table 2. The analysis and comparisons below, with the exception of the case study, are presented in terms of best and average over 10 runs.

The BKS in the literature were obtained on different hardware. Benjamin (2011) uses a 3.16 GHz Intel Core2 Duo machine with 3.23 GB of memory, Buhrkal et al. (2012) use a 2.67 GHz Intel Core i7 machine with 8.00 GB of memory, and Hemmelmayr et al. (2013) use a 2.4 GHz machine with 4 GB of memory, but the processor type is not specified. Moreover, the algorithms are implemented in different languages and some run on different platforms. Thus, scaling of computation times will almost certainly be biased. Therefore, we report the original computation times with the remark that all results were produced on recent processor architectures.

5.1 Evaluation on Randomly Generated Instances

[New results using modified Schneider et al. (2014a) instances to come here.]

5.2 Tests on Benchmark Instances from the Literature

The first set of instances from the literature was developed by Kim et al. (2006). These include 10 instances with a single depot, up to 19 dumps and up to 2092 containers. They have time windows and a mandated break with a time window. We solved the first five instances. In all tables below, instance sizes are specified as (number of containers, number of dumps). The BKS for these instances were obtained by either Benjamin (2011) or Buhrkal et al. (2012). Benjamin's (2011) algorithm is deterministic and she reports only one result per instance, while Buhrkal et al. (2012) report the best and average over 10 runs. Therefore, in Table 3 we report the best result from Benjamin (2011) or Buhrkal et al. (2012), and the average of Buhrkal et al. (2012). Computation times are those of Buhrkal et al. (2012). Benjamin's (2011) computation times are up to two degrees of magnitude longer than those of Buhrkal et al. (2012).

Table 3: Comparison Against the Best Known Solutions to the VRP-IF (Kim et al., 2006) Instances

		BKS from literature			This work				
Inst-	(points,			Runtime			Runtime	Gap	Gap
ance	dumps)	Best	Average	avg(s.)	Best	Average	avg(s.)	best(%)	avg(%)
102	(99,2)	156.90	176.03	2.00	150.66	153.67	217.88	-3.98	-12.70
277	(275,1)	447.60	455.70	8.00	444.24	448.85	2304.57	-0.75	-1.50
335	(330,4)	182.10	196.49	10.00	177.84	180.63	2606.67	-2.34	-8.07
444	(442,1)	78.30	79.00	18.00	77.88	78.42	1893.39	-0.54	-0.73
804	(784,19)	604.10	650.65	72.00	594.13	618.18	3884.68	-1.65	-4.99
Avg				22.80			2181.44	-1.85	-5.60

Table 4: Comparison Against the Best Known Solutions to the MDVRPI (Crevier et al., 2007) Instances

		Hemmelmayr et al. (2013)			This work				
Inst-	(points,			Runtime			Runtime	Gap	Gap
ance	dumps)	Best	Average	avg(s.)	Best	Average	avg(s.)	best(%)	avg(%)
a1	(48,2)	1179.79	1180.57	85.20	1189.18	1202.89	21.12	0.80	1.89
b1	(96,2)	1217.07	1217.07	383.40	1217.07	1231.33	190.62	0.00	1.17
c1	(192,2)	1866.76	1867.96	1224.00	1885.57	1910.21	712.35	1.01	2.26
d1	(48,3)	1059.43	1059.43	94.20	1059.43	1071.19	19.33	0.00	1.11
e1	(96,3)	1309.12	1309.12	373.20	1309.12	1333.99	157.02	0.00	1.90
f1	(192,3)	1570.41	1573.05	1536.00	1576.81	1597.78	1148.62	0.41	1.57
g1	(72,4)	1181.13	1183.32	202.80	1186.59	1202.28	72.50	0.46	1.60
h1	(144,4)	1545.50	1548.61	876.60	1559.21	1571.26	531.82	0.89	1.46
i1	(216,4)	1922.18	1923.52	2014.80	1933.30	1956.97	1224.14	0.58	1.74
j1	(72,5)	1115.78	1115.78	166.80	1119.39	1139.20	66.34	0.32	2.10
k1	(144,5)	1576.36	1577.96	873.60	1581.23	1598.25	555.05	0.31	1.29
l1	(216,5)	1863.28	1869.70	2128.80	1880.93	1903.15	1435.59	0.95	1.79
a2	(48,4)	997.94	997.94	73.80	997.94	998.90	37.81	0.00	0.10
b2	(96,4)	1291.19	1291.19	384.60	1294.77	1343.87	217.86	0.28	4.08
c2	(144,4)	1715.60	1715.84	900.60	1731.60	1756.83	432.03	0.93	2.39
d2	(192,4)	1856.84	1860.92	1808.40	1863.97	1884.91	1031.17	0.38	1.29
e2	(240,4)	1919.38	1922.81	2958.60	1939.02	1979.30	1621.11	1.02	2.94
f2	(288,4)	2230.32	2233.43	4274.40	2273.17	2291.38	2451.33	1.92	2.59
g2	(72,6)	1152.92	1153.17	222.60	1153.21	1167.65	77.96	0.02	1.26
h2	(144,6)	1575.28	1575.28	939.60	1583.12	1601.21	506.46	0.50	1.65
i2	(216,6)	1919.74	1922.24	2515.20	1927.44	1958.01	1402.32	0.40	1.86
j2	(288,6)	2247.70	2250.21	4402.80	2259.99	2291.22	3056.50	0.55	1.82
Avg				1292.73			771.32	0.53	1.81

As shown in Table 3, we obtain significant improvements over the BKS from the literature, but our computation times are slower for several reasons. First, the heuristic is not designed for such problem sizes as they do not appear in our case studies. Secondly, we did not optimize the heuristic parameters for specific instance classes, and all results were produced with the parameter values from Table 2. And thirdly, there is an overhead from the additional features considered in our problem. Since these instances assume an unlimited homogeneous fleet, we also report the number of vehicles used in each instance, respectively 3, 3, 6, 11, and 5, like in the BKS.

Table 4 presents our results for the MDVRPI (Crevier et al., 2007) instances, which include a single vehicle depot, various numbers of intermediate facilities and a maximum tour duration, but no time windows or driver breaks. The BKS are obtained by Hemmelmayr et al. (2013), who use a VNS with

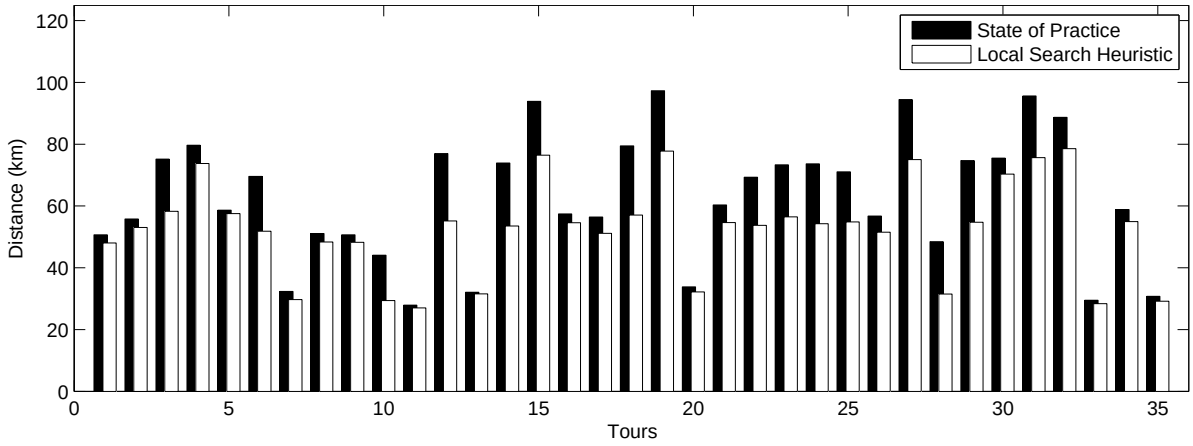
Table 5: Potential Savings from a Flexible Assignment of Destination Depots in the MDVRPI (Crevier et al., 2007) Instances [add an explanatory note]

		Hemmelmayr et al. (2013)			This work				
Inst-ance	(points, dumps)	Best	Average	Runtime avg(s.)	Best	Average	Runtime avg(s.)	Gap best(%)	Gap avg(%)
a1	(48,2)	1179.79	1180.57	85.20	1094.85	1106.46	20.52	-7.20	-6.28
b1	(96,2)	1217.07	1217.07	383.40	1208.23	1218.30	132.22	-0.73	0.10
c1	(192,2)	1866.76	1867.96	1224.00	1851.59	1885.82	764.14	-0.81	0.96
d1	(48,3)	1059.43	1059.43	94.20	1009.14	1023.26	27.05	-4.75	-3.41
e1	(96,3)	1309.12	1309.12	373.20	1280.14	1294.99	147.48	-2.21	-1.08
f1	(192,3)	1570.41	1573.05	1536.00	1544.27	1568.29	945.87	-1.66	-0.30
g1	(72,4)	1181.13	1183.32	202.80	1131.75	1138.56	65.15	-4.18	-3.78
h1	(144,4)	1545.50	1548.61	876.60	1523.97	1542.88	448.36	-1.39	-0.37
i1	(216,4)	1922.18	1923.52	2014.80	1900.70	1936.75	1443.26	-1.12	0.69
j1	(72,5)	1115.78	1115.78	166.80	1076.55	1080.02	68.83	-3.52	-3.21
k1	(144,5)	1576.36	1577.96	873.60	1525.45	1542.00	519.69	-3.23	-2.28
l1	(216,5)	1863.28	1869.70	2128.80	1846.76	1874.47	1249.17	-0.89	0.26
a2	(48,4)	997.94	997.94	73.80	887.58	911.09	45.10	-11.06	-8.70
b2	(96,4)	1291.19	1291.19	384.60	1256.27	1273.99	184.68	-2.70	-1.33
c2	(144,4)	1715.60	1715.84	900.60	1691.70	1715.05	421.25	-1.39	-0.05
d2	(192,4)	1856.84	1860.92	1808.40	1860.77	1870.70	833.90	0.21	0.53
e2	(240,4)	1919.38	1922.81	2958.60	1913.66	1951.95	2016.85	-0.30	1.52
f2	(288,4)	2230.32	2233.43	4274.40	2249.43	2274.70	2472.20	0.86	1.85
g2	(72,6)	1152.92	1153.17	222.60	1070.38	1085.91	119.28	-7.16	-5.83
h2	(144,6)	1575.28	1575.28	939.60	1550.94	1566.95	369.27	-1.54	-0.53
i2	(216,6)	1919.74	1922.24	2515.20	1903.29	1925.68	946.27	-0.86	0.18
j2	(288,6)	2247.70	2250.21	4402.80	2239.79	2271.01	2913.17	-0.35	0.92
Avg				1292.73			734.26	-2.54	-1.37

a dynamic programming procedure for the insertion of the intermediate facilities. Overall, in several cases we reach the BKS and our best solutions have an average gap of 0.53% with respect to the BKS. The gap with respect to the average solutions over 10 runs stands at 1.81%. These instances are also attempted by Benjamin (2011) who reach an average gap of 7.87% from the BKS. We can thus stress on the importance of the more general nature of our approach, which is stable across various instance classes.

[CAREFULL - REWRITE POSSIBLE (FLEXIBLE ASSIGNMENT OF DESTINATION DEPOTS): In order to assess the potential savings from allowing a flexible assignment of destination depots, we relax the MDVRPI instances by considering all intermediate facilities as possible destination depots of any vehicle. It should be noted that in the original MDVRPI formulation, intermediate facilities are actually depots with no vehicles stationed there. We consider the extreme case of a relocation cost term weight Ψ of zero in the objective function. Table 5 demonstrates that important improvements can be obtained. Naturally, using a relocation cost term with a weight between zero and one will lead to results that fall between the restricted case and the extreme case presented in Table 5. In the results we obtained, the improvements over the restricted case are due both to the fact that vehicles can now use any depot for origin and destination and also to the fact that many tours start and end at different depots, thus better exploiting the geographical characteristics of the instances. In a realistic situation where depots are not located in the center of the service area, the benefits may be even more pronounced. The case study area to which this type of tours are applicable is in a French sparsely populated rural area. However, no historical tour data is available from the collector for comparison. The purpose of this test was therefore to justify the approach and quantify the potential benefits using

Figure 3: Distance Improvements Against the Current State of Practice



synthetic instances.]

5.3 Real-world Case Study

The real-world case study considers a Swiss collector of recyclable waste in the Geneva area that uses specialized software to plan its collection tours. The collector is responsible for approximately 750 containers positioned at 250 distinct locations, and the software is currently used for planning some of the tours for collecting white glass and PET. The number of white glass and PET containers is 170 and 140, respectively, and the available fleet consists of six vehicles with varying characteristics, whose weight capacities range from nine to 14 tons. The currently used software can only plan one tour at a time and does not support all features required by the collector and present in our problem definition.

We obtained a sample of 35 planned tours for white glass and PET, their size ranging from seven to 38 containers, and with up to four dump visits per tour. The distances between all depots, containers and dumps are shortest paths on a road network obtained from a geographic information system. To perform a fair comparison between the software currently used for planning the tours and our local search heuristic, we re-solved the problem for each tour separately, only enforcing the supported features, which are limited to vehicles' volume and weight capacity. Moreover, in the sample we obtained, all tours were planned for a different date, and could therefore use the same vehicle or visit the same container. As a consequence, we could not combine multiple tours to be solved as a single instance. To compensate for the lack of richer features in the available real-world data, the latter have been tested in the experiments in Sections 5.1 and 5.2

Figure 3 compares the distances of the tours as planned by the software currently used by the collector and as proposed by our local search heuristic. The results of our heuristic are averaged over 10 runs and computation times range from 0.05 to 7.58 s., with an average of 1.21 s. As the figure shows, all tours are improved and average improvement per instance ranges from 1.73% to 34.91%, with a mean of 14.64%. It is interesting to observe that the distance improvements are due both to better container sequencing and better planned visits to the available recycling facilities.

After consultations with the concerned company, we can estimate direct financial savings from fuel and labor in the order of 300,000 USD annually. These estimations assume that the number of tours is kept unchanged. However, given that the proposed solution approach can optimize multiple tours at the same time, rather than one at a time as in the current state of practice, further savings from a

reduced number of tours, better planning of dump visits and more efficient labor utilization can also be expected. To give a better idea of the scale of the savings, we remark that the company in question is just one of several in a canton of 480,000 inhabitants.

It should be mentioned here that the software currently used by the collector requires a validation step once the planned tour has been executed. During validation, the collector deletes those containers from the originally proposed sequence that for some reason were not collected during the tour. However, the ordering of the sequence cannot be changed. We use our heuristic to build tours for the containers in the validated sequences and compare to their travel distances. The originally proposed sequences before validation are not available, but fortunately, not all validated tours have had containers removed, and those that have usually have one or two containers removed.

Thus the results from Figure 3 provide strong evidence in favor of our local search heuristic as compared to the current state of practice. Moreover, our heuristic is capable of solving a much richer problem and for a fraction of the computation time.

6 Conclusion

This article proposes a mathematical model and a local search heuristic for a complex solid waste collection problem, an extension of the VRP-IF with a heterogeneous fixed fleet and a flexible assignment of destination depots. We include several additional side constraints, such as a mandated break period contingent on tour start time, multiple vehicle capacities and site dependencies, and consider a general cost function corresponding to the cost structure of a typical firm.

The extensive computational testing we performed shows that the heuristic achieves optimality on small random instances, exhibits competitive performance in comparison to state-of-the-art solution methods for special cases of our problem, and leads to important savings in the state of practice. We demonstrated that the flexibility in destination depot assignment can lead to noticeable savings if it is properly understood and optimized, which is not the case for most waste collectors, especially in rural and sparsely populated areas where such benefits will be most pronounced. Another benefit of our solution approach is its short computation times for the size of the currently observed real-world instances. Moreover, it outperforms significantly the solution currently in place in terms of quality and functionality.

The evaluation of the heuristic on multi-tour real-world instances using a heterogeneous fixed fleet remains a task for the future, when more detailed historical performance records will be available as the system is currently being deployed to more collectors. The heuristic implementation remains to be integrated into the business processes of a Swiss software-as-a-service and infrastructure management provider of waste collection logistics solutions. The flexibility and robustness of the methodology will thus be subject to real-world testing and assessment. Future extensions of the solution approach will also see the development of a non-linear forecasting model for the container filling rates based on sensor data and its integration into an inventory routing solution with more sophisticated search algorithms. Given the applied nature of our problem, the inclusion of detailed European driver regulations presents opportunities for further useful extensions of the solution algorithm.

Acknowledgments

This work is partially funded by Switzerland's Commission for Technology and Innovation (CTI) under grant number CTI 15781.1 PFES-ES. The authors would also like to thank EcoWaste SA, industrial partners under this grant, for their collaboration, expert advice and discussion on industry and problem specific issues, and the real data they provided for the experiments in this study.

References

Angelelli, E. and Speranza, M. G. (2002a). The application of a vehicle routing model to a waste-collection problem: Two case studies. *The Journal of the Operational Research Society*, 53(9):944–952.

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

Baldacci, R. and Mingozzi, A. (2009). A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming*, 120(2):347–380.

Bard, J. F., Huang, L., Dror, M., and Jaillet, P. (1998a). A branch and cut algorithm for the VRP with satellite facilities. *IIE Transactions*, 30(9):821–834.

Bard, J. F., Huang, L., Jaillet, P., and Dror, M. (1998b). A decomposition approach to the inventory routing problem with satellite facilities. *Transportation Science*, 32(2):189–203.

Beltrami, E. J. and Bodin, L. D. (1974). Networks and vehicle routing for municipal waste collection. *Networks*, 4(1):65–94.

Benjamin, A. M. (2011). *Metaheuristics for the Waste Collection Vehicle Routing Problem with Time Windows*. PhD thesis, Department of Mathematical Sciences, Brunel University, London, UK.

Buhrkal, K., Larsen, A., and Ropke, S. (2012). The waste collection vehicle routing problem with time windows in a city logistics context. *Procedia-Social and Behavioral Sciences*, 39:241–254.

Coene, S., Arnout, A., and Spieksma, F. C. R. (2010). On a periodic vehicle routing problem. *Journal of the Operational Research Society*, 61:1719–1728.

Conrad, R. G. and Figliozzi, M. A. (2011). The recharging vehicle routing problem. In Doolen, T. and Aken, E. V., editors, *Proceedings of the 2011 Industrial Engineering Research Conference*, Reno, NV, USA.

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

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

Erdoğan, S. and Miller-Hooks, E. (2012). A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100–114. Select Papers from the 19th International Symposium on Transportation and Traffic Theory.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

Golden, B., Assad, A., Levy, L., and Gheysens, F. (1984). The fleet size and mix vehicle routing problem. *Computers & Operations Research*, 11(1):49–66.

Hemmelmayr, V., Doerner, K. F., Hartl, R. F., and Rath, S. (2013). A heuristic solution method for node routing based solid waste collection problems. *Journal of Heuristics*, 19(2):129–156.

Hemmelmayr, V. C., Doerner, K. F., Hartl, R. F., and Vigo, D. (2014). Models and algorithms for the integrated planning of bin allocation and vehicle routing in solid waste management. *Transportation Science*, 48(1):103–120.

Kek, A. G., Cheu, R. L., and Meng, Q. (2008). Distance-constrained capacitated vehicle routing problems with flexible assignment of start and end depots. *Mathematical and Computer Modelling*, 47:140–152.

Kim, B. I., Kim, S., and Sahoo, S. (2006). Waste collection vehicle routing problem with time windows. *Computers & Operations Research*, 33(12):3624–3642.

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

Ombuki-Berman, B. M., Runka, A., and Hanshar, F. T. (2007). Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms. In *Proceedings of the Third IASTED International Conference on Computational Intelligence*, CI '07, pages 91–97, Anaheim, CA, USA.

Penna, P. H. V., Subramanian, A., and Ochi, L. S. (2013). An iterated local search heuristic for the heterogeneous fleet vehicle routing problem. *Journal of Heuristics*, 19(2):201–232.

Prescott-Gagnon, E., Desaulniers, G., and Rousseau, L.-M. (2014). Heuristics for an oil delivery vehicle routing problem. *Flexible Services and Manufacturing Journal*, 26(4):516–539.

Rochat, Y. and Taillard, É. D. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1:147–167.

Sahoo, S., Kim, S., Kim, B.-I., Kraas, B., and Popov, A. (2005). Routing optimization for waste management. *Interfaces*, 35(1):24–36.

Savelsbergh, M. W. P. (1992). The vehicle routing problem with time windows: Minimizing route duration. *INFORMS Journal on Computing*, 4(2):146–154.

Schneider, M., Stenger, A., and Goeke, D. (2014a). The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science*, 48(4):500–520.

Schneider, M., Stenger, A., and Hof, J. (2014b). An adaptive VNS algorithm for vehicle routing problems with intermediate stops. *OR Spectrum* (published online), pages 1–35.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265.

Subramanian, A., Penna, P. H. V., Uchoa, E., and Ochi, L. S. (2012). A hybrid algorithm for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, 221(2):285–295.

Taillard, É. D. (1999). A heuristic column generation method for the heterogeneous fleet VRP. *RAIRO - Operations Research*, 33:1–14.

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