Efficient Heuristics for Real-world Large-scale Waste Collection Problems

Wenxing Lan, Ziyuan Ye, Peijun Ruan, Jialin Liu, *Senior Member, IEEE,* and Xin Yao, *Fellow, IEEE*

***Abstract*—Waste collection is a common yet essential problem in real world. In this work, we consider a large-scale waste collection problem with many constraints that no existing work has ever solved such problem of this size. The studied problem has multiple depots and disposal facilities. Vehicles with limited capacity can make multiple trips to different disposal facilities, but they should start from and end to an identical depot at the beginning and end of a day, respectively. Each vehicle has a working time constraint, which means that it cannot keep collecting all day long. In this paper, we formalise this complex routing problem with all the above-given constraints, aiming at minimising the total mileage of vehicles. We also propose a simple yet efficient heuristic-assisted solution initialisation algorithm for generating solutions of high quality. Then, the initial solution is further optimised by our heuristic-assisted extended local search consisting of three new search operators. Our approaches have been evaluated on a real-world problem with thousands of tasks and shown superior performance comparing with Clarke and Wright savings algorithms and sophisticated memetic algorithm, respectively. The proposed approaches can be extended to other routing problems with the same number of or fewer constraints.**

***Index Terms*—Capacitated vehicle routing problem, waste collection, large-scale, multi-depot, multi-trip, multi-disposal-facility.**

I. Introduction

THE capacitated vehicle routing problem (CVRP) is a classic NP-hard combinatorial optimisation problem aiming at efficiently allocating several vehicles with limited capacity to serve different customers under some constraints [1]–[3]. In recent years, numerous variants of CVRP considering different constraints have been derived and researched. Examples include, but not limited to, the capacitated vehicle routing problem with time-windows (CVRPTW) [4], the multi-depot capacitated vehicle routing problem (MDCVRP) [5], and the multi-trip capacitated vehicle routing problem (MTCVRP) [6]. The CVRP and its variants have many inspiring applications in real life, such as green logistics [7], drone delivery [8] and waste collection [9].

The waste collection problem (WCP) is an important and common real-world problem, which focuses on efficiently collecting and dumping waste in a city [10]. Examples of the waste collection process are shown in Figure 1. The WCP in real-life is challenging due to the numerous constraints defined by the multiple depots (parking lots), the multiple disposal facilities, the limited capacity of vehicles, the limited working duration, as well as the large number of collection sites. In particular, with the continuous expansion of city size and increase of population, the coverage area of waste collection in the city becomes larger and larger. Therefore, it is necessary to schedule the routes reasonably and effectively to reduce the total mileage of vehicles, the working time and even the number of allocated vehicles, i.e., different types of costs.

WCP has been formalised as different variants of CVRP, taking into account one of the following constraints alone: multiple depots [11], multiple intermediate disposal facilities (inter-depots) [12] and multiple trips [13]. However, to the best of our knowledge, none of the existing work has considered multiple depots, multiple disposal facilities and multiple trips at the same time. Thus, the methods and techniques designed for those existing models are not directly applicable to the large-scale multi-depot multi-disposal-facility multi-trip capacitated vehicle routing problems (M3CVRP) in real life. To fill the gap between the current research and the need of real-world applications, in this paper, we propose M3CVRP (Section III-B), a new and more realistic model, and perform a case study on a real-world problem with $3,000$ collection sites inside and around the Qingdao city of China (Section III-A).

The features of the real-world WCP studied in this paper (detailed in Section III) are summarised as follows.

(i) *Large-scale*: $3,000$ collection sites are considered. As a result, many traditional methods for small datasets will no more be suitable or take too much time to find a satisfactory solution.
(ii) *Multi-depot*: for the practical issue, each collection site can be recycled by waste vehicles from different depots.
(iii) *Multi-disposal-facility*: a fully-loaded vehicle can dump in different disposal facilities.
(iv) *Multi-trip*: vehicles can carry out the process of waste collection, transportation, and dumping multiple times before returning to the depot.
(v) *Limited maximum working time*: vehicles cannot keep collecting all day long.

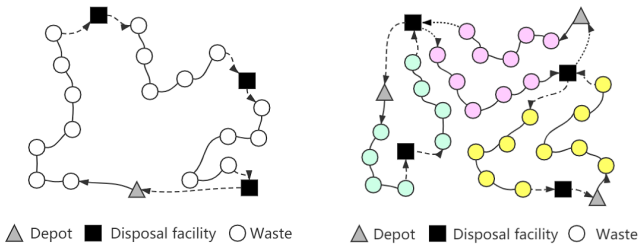
Figure 1b demonstrates an illustrative example of a multi-depot multi-disposal-facility multi-trip route.

The authors are with the Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China.

W. Lan, Z. Ye and P. Ruan contributed equally to this work.

Corresponding author: Jialin Liu (liujl@sustech.edu.cn).

This work was supported by the National Key R&D Program of China (Grant No. 2017YFC0804003), the National Natural Science Foundation of China (Grant No. 61906083, 61976111), the Guangdong Provincial Key Laboratory (Grant No. 2020B121201001), the Program for Guangdong Introducing Innovative and Enterpreneurial Teams (Grant No. 2017ZT07X386), the Shenzhen Science and Technology Program (Grant No. KQTD2016112514355531), the Science and Technology Innovation Committee Foundation of Shenzhen (Grant No. JCYJ20190809121403553) and the Program for University Key Laboratory of Guangdong Province (Grant No. 2017KSYS008).



(a) Trips considering multiple disposal facilities with one single depot. The dashed lines refer to the paths from waste collection sites to the closest disposal facilities and vice versa.

(b) Trips considering multiple disposal facilities with multiple depots. Different types of dashed lines refer to the paths belonging to different routes.

Fig. 1. Examples of routing plans for vehicles. The solid lines refer to the paths between two different waste collection sites or the paths from the depot to the closest waste collection site.

The considered WCP is confronted with more challenges due to being closer to real-life scenarios. The multi-depot, multi-trip and multi-disposal-facility features result in the requirements of complex considerations with more constraints. Moreover, a large number of collection sites leads to an increase of search space which consumes more time for evaluating solutions and searching for optimal solutions.

To tackle these extremely complex features and challenges presented above, new problem model and efficient heuristics are needed to boost the optimisation. The main contribution of this work as as follows.

(i) We formalise a more realistic model for real-world waste collection problem, namely M3CVRP (Section III-B).
(ii) We propose a novel heuristic-assisted solution initialisation algorithm (HaSI) for generating solutions of high quality (Section IV-C).
(iii) We also design an extended local search (ELS) as an efficient optimiser, the core components of which are two region-constrained swap operators and a relaxed multi-point swap operator (Section V-C).
(iv) In particular, a new disposal facility (inter-route depots) insertion with the backspacing method, which further improves the mileage when completing solutions for evaluation (Section V-B), is proposed.

To the best of our knowledge, no one has ever considered such a large-scale problem, we have given a direction to deal with such issues. Our proposed approach can be extended to other routing problem with the same number of or fewer constraints.

The rest of this paper is organised as follows. Section II discusses the related work. Section III describes the real-world WCP and gives the mathematical formulation of M3CVRP. The details of proposed HaSI and corresponding experimental study are presented in Section IV, while Section V presents our ELS and corresponding experimental study. Finally, Section VI concludes and points out some potential future directions.

II. Background

A. Related Work

Dantzig and Ramser [1] first introduced the vehicle deliver problems and modelled such problems as CVRP. However, a general CVRP is a simplified model and many constraints in real life are not considered. At the early stage of research around CVRP, exact methods were often used to search for optimal solutions [4], [14]. Nevertheless, almost all exact methods are only suitable for optimising problems of small size and suffer from the curse of dimensionality [15], [16]. The search space of CVRP increases exponentially when the problem dimension increases. Exact methods can hardly find optimal solutions in an acceptable time when the problem size is huge. Hence, heuristic methods are widely used to solve CVRPs approximately [17]–[19].

Beltrami and Bodin [10] applied the Clarke and Wright (CW) algorithm [20] to solve waste collection problems where the tasks are located on nodes. Pichpibul and Kawtummachai [21] improved the CW algorithm by changing traverse order of savings list with the mechanism of tournament and the roulette wheel selection. Meta-heuristics have also been applied. In the work of Mi *et al.* [22], the nearest neighbour's algorithm (NNA) was implemented to generate initial solutions for genetic algorithm (GA). Kim *et al.* [9] focused on solving large-scale waste collection problems with one single depot and $2,100$ tasks by a capacitated clustering-based waste collection VRPTW algorithm. He and Liu [12] divided a multi-depot VRP into several sub-problems through clustering, and then solved each sub-problem by an ant colony separately. Memetic algorithm (MA), first introduced by Moscato *et al.* [23], is another popular meta-heuristic for solving VRPs. Minh *et al.* [24] adapted MA with partial-mapped crossover operator to generate offspring and λ-interchange operator in local search. Akhtar *et al.* [16] focused on solving the dynamic urban waster collection problem in which the volume of waste at each collection site can be detected in real-time through sensors. Benjamin and Beasley [25] noticed the perturbation to the quality of solution caused by the insertion of disposal facilities, especially when multiple trips occur. A disposal facility positioning (DFP) algorithm was proposed to search for proper positions for inserting the disposal facilities [25].

The work reviewed previously modelled the WCPs as variants of CVRP. WCP has also often been modelled as variants of capacitated arc routing problem (CARP) [16], [22], [26]–[29]. Decomposition methods have been designed for handling large-scale CARPs, such as the random route grouping (RGG) and the route distance grouping (RDG) [27]. The variable neighborhood decomposition (VND) [28] was implemented within the cooperative co-evolution (CC) framework so that the decomposition can be adjusted dynamically. More recently, a scalable approach based on hierarchical decomposition (SAHiD) was proposed by Tang *et al.* [29].

Most of the existing work on CVRP and CARP focused on solving problems with dozens of collection sites, while the number of collection sites in our case is much higher. Kim *et al.* [9] solved a vehicle routing problem with time windows (VRPTW) with $2,100$ collection sites but still far

fewer than ours. Therefore, existing algorithms may not be directly applicable to our large-scale problem and the existing decomposition methods may fail due to a large number of tasks to be clustered. Only SAHiD was applied to a problem of 3,584 tasks [29]. Nevertheless, how to apply SAHiD to our real-world UWCP is not evident as the SAHiD was designed for CARP with one single depot (which is at the same time a disposal facility), while in our case, multiple depots and multiple disposal facilities are involved.

B. Classical Search Operators in Local search

Search operators is one of the core components of local search algorithms. A good number of search operators have been designed for routing problems with different constraints. Some popular search operators, such as insertion, swap and 2-opt [30], have been widely extended [31]–[36]. Song and Dong [37] adopted swap and 2-opt to solve CVRP by embedding local search (LS) with these two search operators into differential evolution (DE) algorithm. The CROSS-exchange, first proposed by [38], was adapted to swap segments of consecutive waste collection sites between two routes, and was embedded with tabu search to solve the vehicle routing problem with soft time windows (VRPSTW) [39]. Kim *el al.* applied CROSS-exchange to single routes in simulated annealing (SA) algorithm in order to improve the sub-route construction performance of the waste collection vehicle routing problem with time windows (VRPTW) [9]. GENICROSS operator, which draws upon GENIUS insertion heuristic [40] and CROSS-exchanges, and IOPT-operator [41] were used to optimise VRPTW [41]. Three neighbourhood operators, denoted as N_1, N_2 and N_3, were adopted by Zhou and Wang to solve multi-objective vehicle routing problem with time windows [42]. Particularly, N_1, similar to insertion operator, randomly selects a waste collection site from a route and re-inserts it into the best position of the same route [42]. The difference between N_2 and N_1 is that N_2 removes a random number of waste collection sites rather than one and N_3 exchanges a sequence of sites with the same orientation and the same order [42]. The search operators mentioned above except the ones in [42] may be not suitable to large-scale M3CVRP. Since these search operators with greater randomness cannot obtain better solutions efficiently.

III. Real-world Waste Collection Problem

A. Problem Description

Our case study is based on a real-world WCP of $3,000$ collection sites, 3 disposal facilities and 3 depots located inside and around the Qingdao city of China. It is worth mentioning that one of the depots is at the same time a disposal facility. Thus, there are $3,005$ locations (or nodes) in total. Figure 2 visualises the distribution of locations. As a realistic problem, the dataset of this problem is extremely large, which consists of thousands of waste collection sites, multiple depots and multiple disposal facilities. It is not trivial to solve such a complex problem. As the problem grows in size and complexity, the size of the solution search space grows exponentially, and the difficulty of finding a good solution also increases significantly. In addition, the WCPs with several depots and several disposal facilities have rarely been discussed in existing work (cf. Section II).

A vehicle is allowed to collect waste at one or more collection sites under the condition that its load does not exceed its capacity. A vehicle can empty its load to any disposal facility, and then start a new trip, which will lead to multiple trips to disposal facilities or return to its depot. The aim of solving WCP is to find a set of routes that satisfies the constraints listed below.

- $\mathcal{C}_1$: Each vehicle starts and ends at the same depot at the beginning and end of the day, respectively.
- $\mathcal{C}_2$: Each waste collection site should be served exactly once by exactly one vehicle.
- $\mathcal{C}_3$: The load of any vehicle at any moment cannot exceed its capacity.
- $\mathcal{C}_4$: No vehicle is allowed to work longer than its legal working time.
- $\mathcal{C}_5$: The load of any vehicle leaving a depot or a disposal facility should be 0.

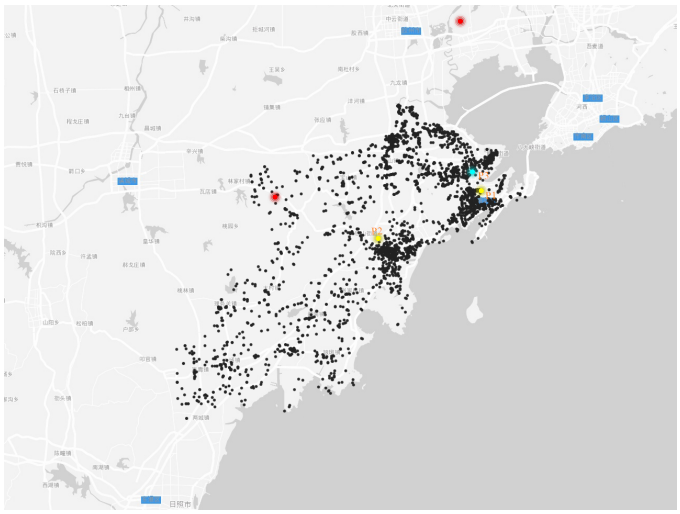


Fig. 2. The visualisation of depots (red points denoted as P_1, P_2 and P_3), waste collection sites (black points) and disposal facilities (yellow points) on real-world map. The cyan-blue point refers to the depot that is also a disposal facility.

1) Data Collection and Pre-processing: The identity, volume of waste, location (latitude and longitude) and service cost of each collection site are given in the dataset, as well as the name, identity and location of depots and disposal facilities. The service cost of each disposal facility is set as 0 since there is no such data in the original dataset. The distance from any location A to location B, denoted as $dist(A, B)$, was not given in the original dataset, but determined as the distance of driving route from A to B calculated by a public API of *AutoNavi*[1]. Note that $dist(A, B)$ does not always equal to $dist(B, A)$ due to single lanes. Previous work rarely considered such kind of situation.

[1]https://lbs.amap.com/api/javascript-api/guide/services/navigation

2) Necessity of a New Model: Compared with existing work on VRPs, the characteristics of our model include multiple depots, multiple disposal facilities and the limitation on the working time of each vehicle. To the best of our knowledge, VRPs with multiple disposal facilities (inter-route depots) are rarely considered. Most work only considered one single disposal facility, which is, at the same time, the only depot in their dataset [16], [22], [29], [43]. Multiple trips are usually caused due to multiple disposal facilities, and make the problem more complicated as it is necessary to consider the selection and insertion of disposal facilities in a route [9], [44], [45]. Existing solutions to tackle the problem with multiple depots usually consider converting it into problems with single depot by clustering or classification of collection sites [9], [46].

The issues and drawbacks of applying directly existing models to our real-world problem are summarised below.

- Scalability issue: Most of the current research work on small- or middle-size datasets which contain fewer than $1,000$ points. When dealing with large datasets, the issue of the dimensional explosion may happen. For instance, when using CPLEX to solve TSP, the memory required increases exponentially with the increase of the number of points, owing to the fact that the number of decision variables and constraints has significantly increased [47], [48]. Thus, if using CPLEX for solving our problem with $3,000$ sites, at least 12TB of memory and several days of runtime are required to find a feasible solution, which is not acceptable in the real-world situation.
- Unpractical Euclidean distance: Actual travelling distance between locations is rarely considered in existing work, while Euclidean distance is mostly used [9], [49], [50]. However, in real life, two points with short Euclidean distance may be very far apart, as, for example, there is a river between these two points.
- Legal working time: In real-world scenarios, the maximum working time of vehicles should be taken into account. In other words, no vehicle is permitted to collect and transport waste indefinitely [51].
- Multiple depots and multiple disposal facilities: For large cities (e.g., Shenzhen and Los Angeles), there are usually many disposal facilities and depots. With the acceleration of urbanisation, the model of single depot or single disposal facility [48], [52] will become less and less applicable.

Therefore, a new problem model considering all above is necessary.

B. M3CVRP

To handle the WCP described above, we formalise a novel mathematical model, multi-depot multi-trip multi-disposal-facility capacitated vehicle routing problem (M3CVRP). The city and surrounding area can be regarded as a directed graph $G = (V, E)$, where V is the vertex set and E is the edge set. The vertex set V is the union set of the set of depots (V_P), the set of waste collection sites (V_C), and the set of disposal facilities (V_D). It is notable that in our real-world WCP, one of the disposal facility is also a depot which is often the case for other cities, i.e., $V_P \cap V_D \neq \emptyset$ and $V_P \neq V_D$. Each collection site $c \in V_C$ has a non-negative amount of waste $t(c)$ to be collected and a non-negative service time $\delta(c)$. Disposal facilities and depots have no waste and no service time is consumed in depots in our practical application, so the $t(p)$ is set as 0 for $p \in V_P \cup V_D$, and $\delta(p)$ is set as 0 for $p \in V_P$. Every disposal facility has an infinite capacity which means that $\forall d \in V_D$ can be visited for an infinite number of times. The edge set E is defined as $\{(i,j) \mid \forall i,j \in V, i \neq j\}$ and the distance from vertex i to vertex j is denoted by $dist(i,j)$. M denotes the number of vehicles. Each vehicle v has a limited capacity $Capacity(v)$, a moving speed $Speed(v)$, and maximum legal hours of work in a day $WT(v)$. Furthermore, $Load(v,i)$ indicates that the load of v at vertex i, where $i \in V$.

We formulate each solution S as a set of distinct routes $\{R_1, \ldots, R_M\}$, where R_k is the route served by the vehicle k, $\forall k \in \{1, \ldots M\}$. Each R_k is associated to one and only one depot denoted by p_k, which implies constraint $\mathcal{C}_1$. Each route consists of several trips and is formalised as $R_k = \{r_{k,1}, \ldots, r_{k,l_k}\}$, where l_k is the number of trips in route R_k. One or more disposal facilities inserted to the route R_k are referred to as $d_{k,1}, \ldots, d_{k,l_k}$. It is possible that a disposal facility is visited more than once in a route or only one disposal facility is visited in a route. The i^{th} trip served by the k^{th} vehicle, $r_{k,i}$, can be represented as a sequence $(c_{k,i,1}, \ldots, c_{k,i,h_{k,i}})$, where $c_{k,i,1}, \ldots, c_{k,i,h_{k,i}} \in V_C$ are the collection sites allocated to this trip and $h_{k,i}$ is the number of served collection sites in this trip. $Set_c(r_{k,i})$ denotes the set of collection sites in $r_{k,i}$. Therefore, $|Set_c(r_{k,i})| = h_{k,i}$ as each site is collected exactly once. A vehicle has an empty trip set if no task has been assigned to it. The shortest path between any pair of vertices is assumed to be available or calculable. The distance of the i^{th} trip of vehicle k, including travelling from/to the disposal facilities, is calculated as

$$\begin{aligned} C_t(r_{k,i}) =& dist(d_{k,i-1}, c_{k,i,1}) + \sum_{j=2}^{h_{k,i}} dist(c_{k,i,j-1}, c_{k,i,j}) \\ &+ dist(c_{k,i,h_{k,i}}, d_{k,i}), \end{aligned} \tag{1}$$

where $dist(d_{k,0}, c_{k,1,1}) = 0$. Then, the distance of the route of vehicle k, including travelling from/to the depot p_k, is calculated as

$$C_r(R_k) = dist(p_k, c_{k,1,1}) + \sum_{i=1}^{l_k} C_t(r_{k,i}) + dist(d_{k,l_k}, p_k). \tag{2}$$

If d_{k,l_k} is identical to p_k, thus the visited disposal facility is also a depot, then $dist(d_{k,l_k}, p_k) = 0$. The objective of M3CVRP is minimising the total travel distance of all allocated vehicles, formalised as (3). The constraint $\mathcal{C}_1$ (described in Section III-A) is implied in the objective function (3). The constraint $\mathcal{C}_2$ is ensured by (4) and (5). The constraints $\mathcal{C}_3$, $\mathcal{C}_4$ and $\mathcal{C}_5$ are explicitly ensured by (6), (7) and (8), respectively.

In this M3CVRP model, multiple depots, multiple disposal facilities, multiple trips, the working hours of each vehicle, the service time of each site and capacity constraints are simultaneously considered, which makes the model extraordinarily complicated but closer to a real-world situation.

$$\min \ Cost(S) = \sum_{k=1}^{M} C_r(R_k), \tag{3}$$

$$\begin{aligned}
S.t. \quad & p_k \in V_D, \forall k \in \{1, \ldots, M\}, \\
& d_{k,i} \in V_D, \forall k \in \{1, \ldots, M\}, \forall i \in \{1, \ldots, l_k\}, \\
& c_{k,i,j} \in V_C, \forall k \in \{1, \ldots, M\}, \forall i \in \{1, \ldots, l_k\}, \forall j \in \{1, \ldots, h_{k,i}\}, \\
& \cup_{k=1}^{M} \cup_{i=1}^{l_k} Set_c(r_{k,i}) = V_C, \qquad (4) \\
& \sum_{k=1}^{M} \sum_{i=1}^{l_k} |Set_c(r_{k,i})| = |V_c|, \qquad (5) \\
& \sum_{j=1}^{h_{k,i}} t(c_{k,i,j}) \leq Capacity(k), \forall k \in \{1, \ldots, M\}, \forall i \in \{1, \ldots, l_k\}, \qquad (6) \\
& \frac{C_r(R_k)}{Speed(k)} + \sum_{i=1}^{l_k} \left(\sum_{j=1}^{h_{k,i}} \delta(c_{k,i,j}) + \delta(d_{k,i}) \right) \leq WT(k), \forall k \in \{1, \ldots, M\}, \qquad (7) \\
& Load(k, p_k) = 0, \forall k \in \{1, \ldots, M\}, p_k \in V_P. \qquad (8)
\end{aligned}$$

IV. Smart Initialisation

In this section, we propose a heuristic-assisted approach (Section IV-C) to generate initial solutions of high quality efficiently. The representation of the initial solution is given in Section IV-A. An adaptation of Clarke and Wright Savings Algorithm (CWSA) (Section IV-B) [20] is employed as a baseline algorithm in the initial solution construction. Experimental study and discussion are presented in Section IV-D and Section IV-E, respectively.

A. Solution Representation

An initial solution is represented as an ordered permutation of tasks, depots and disposal facilities. For instance, the route of vehicle k is presented as: $p_k \rightarrow c_{k,1,1} \rightarrow c_{k,1,2} \rightarrow \cdots \rightarrow c_{k,1,h_{k,l}} \rightarrow d_{k,1} \rightarrow c_{k,2,1} \rightarrow \cdots \rightarrow c_{k,l_k,h_{k,l_k}} \rightarrow d_{k,l_k} \rightarrow p_k$.

B. M3CWSA: An Adaptation of Clarke and Wright Savings Algorithm for M3CVRP

The Clarke and Wright Savings Algorithm (CWSA, Algorithm 1) [21] is arguably the most widely applied heuristic for solving CVRP thanks to its simplicity of implementation and efficiency of calculation [20], [21]. The core idea of CWSA is maximising the distance saved by inserting sites into routes with savings formula. The CWSA has also been used to construct initial solutions for VRPs [53], [54]. The CWSA was designed for routing problems with a single depot, single trip, and symmetrical distance. Therefore, we design M3CWSAs which include M3CWSA-I and M3CWSA-II by adapting CWSA to M3CVRP.

In the following description, the $demand$ denotes the total volume of waste in the collection sites, and $ability$ denotes the total capacity of the available vehicles. We consider three possible scenarios. This method takes all the possible cases that may occur in the M3CVRP into account:

(i) $0 < demand < ability$;

Algorithm 1 Clarke and Wright Savings Algorithm [21].

Input: A set of customers C, a depot p and their locations

1: Initial savings list $\mathcal{L}$ as empty
2: **for** $\forall$ customers i, $j \in C$ **do**
3: Calculate the Euclidean distance $d_{i,j}$ between i and j
4: Calculate the savings value between i and j according to $s_{i,j} = d_{p,i} + d_{j,p} - d_{i,j}$
5: Add $(s_{i,j}, (i,j))$ into $\mathcal{L}$
6: **end for**
7: $\mathcal{L}$ is sorted in the decreasing order of savings value
8: **for** $(s_{i,j}, (i,j)) \in \mathcal{L}$ **do**
9: Include link (i,j) in a route if no route constraint (proposed by [20]) will be violated through the inclusion of the customer i and j in that route
10: **end for**
11: **for** $\forall$ non-routed customers $c \in C$ **do**
12: Set its route as $p \rightarrow c \rightarrow p$
13: **end for**

(ii) $ability \leq demand < 2ability$;
(iii) $demand \geq 2ability$.

The criterion used to determine the actual case is the comparison between the total volume of waste in collection sites that have not been collected ($demand$) and the total capacity of the available vehicles ($ability$). Four sub-problems extracted from these cases and corresponding savings formulas are listed as follows, assuming to serve two different collection sites i and j on the same trip.

$\mathcal{P}_1$ Single trip routing plan from a depot (p) to a disposal facility (d):

$$s_{i,j} = dist(p,j) + dist(i,d) - dist(i,j). \tag{9}$$

$\mathcal{P}_2$ Single trip routing plan from a disposal facility (d_1) to a disposal facility (d_2):

$$s_{i,j} = dist(d_1,j) + dist(i,d_2) - dist(i,j). \tag{10}$$

$\mathcal{P}_3$ Single trip routing plan from a disposal facility (d_1) to another disposal facility (d_2) and end in depot (p):

$$s_{i,j} = dist(d_1, j) + dist(i, d_2) + dist(d_2, p) - dist(i, j). \quad (11)$$

$\mathcal{P}_4$ Single trip routing plan from a depot (p) to a disposal facility (d) and end in the same depot:

$$s_{i,j} = dist(p, j) + dist(i, d) + dist(d, p) - dist(i, j). \quad (12)$$

Specifically, d_1 and d_2 can be the same in our work.

The combination of the above four sub-problems can handle all cases which may happen in M3CVRP. It is thus desirable to propose two different algorithms, Algorithms 2 and 3, for initialising solutions of M3CVRP. Each algorithm has its pros and cons shown in Table I. Algorithm 2 has a more detailed division of possible scenarios which considers the distances between depots and facilities. Algorithm 3 does not consider those distances but is more straightforward and easy to implement.

Note that the details of route constraints in M3CWSAs are different from the ones mentioned in CWSA (Algorithm 1). For example, it is assumed that there are two collection sites i and j where i is already in a route and j has not been assigned to any route. If M3CWSAs want to add an edge (i, j) to a route, i needs to be the last collection site in the traversal of the collection site and not in the interior of that route. The distance measure used in CWSA is symmetrical Euclidean distance. However, M3CWSAs use asymmetrical real-world distances. Consequently, if the above additional constraint is not considered, the calculated savings value with link (i, j) will be different from the savings value obtained from the saving formulas (9)-(12). Likewise, a similar thing will happen to j, if j is in a route and i has not been assigned to any route. Specifically, the two scenarios mentioned above need to be considered at the same time when both i and j have already been included in two different existing routes.

C. Heuristic-assisted Solution Initialisation (HaSI)

This section describes our novel heuristic-assisted solution initialisation approach, which determines the depots of a solution before generating a permutation. A tight approach was applied to optimise the solutions and make the solutions more diverse.

1) Depot-directed route initialisation: Before initialising any individual, every collection site is allocated to its closest depot. Thus, a subset of collection sites $V_c^{(i)} \subset V_c$ is determined for the i^{th} depot ($\forall i \in \{1, \ldots, |V_D|\}$). When generating a solution, the algorithm always selects the depot with the highest number of assigned *unserved* collection sites to start with. If there is no available vehicle in this depot, the depot with the second-highest number of assigned *unserved* collection sites will be chosen and so on. The closest unserved collection site of this depot will be served first. Then, repeatedly choosing the closest unserved collection site for service until there is no enough capacity or the working time constraint is met. The above-proposed selection rule in this research is called *closest-serve-first*.

Algorithm 2 M3CWSA-I, proposed by us.

Input: A set of available vehicles V, a set of waste collection sites C, a set of disposal facilities D and a set of depots P

Compute $demand = \sum_{c \in C} t(c)$
Compute $ability = \sum_{v \in V} Capacity(v)$
if $demand < ability$ **then**
    Apply $\mathcal{P}_4$ to all available vehicles until no collection site can be merge into any route
else if $demand \leq ability < 2demand$ **then**
    Divide the available vehicles into two sets V_A and V_B satisfying $\sum_{v \in V_A} Capacity(v) > 2demand - ability$
    Apply $\mathcal{P}_4$ to all vehicles in V_B
    Apply $\mathcal{P}_1$ and then $\mathcal{P}_3$ to all vehicles in V_a
else
    Apply $\mathcal{P}_1$ to all vehicles in V
    while the number of unserved sites > 0 **do**
        Update vehicle information and get currently available vehicles
        if no available vehicles **then**
            break
        end if
        Compute $demand$ and $ability$ for available vehicles
        if $demand \leq ability < 2demand$ **then**
            Divide the available vehicles into two sets V_A and V_B satisfying

$$\sum_{v \in V_A} Capacity(v) > 2demand - ability$$

            Apply $\mathcal{P}_3$ to all vehicles in V_B
            Apply $\mathcal{P}_2$ and then $\mathcal{P}_3$ to all vehicles in V_a
        else
            Apply $\mathcal{P}_2$ to all available vehicles
        end if
    end while
end if
for any unassigned collection site c **do**
    Construct route: $p \rightarrow c \rightarrow d \rightarrow p$ with

$$p, d = \arg\min_{\forall p \in P, \forall d \in D} dist(p, c) + dist(c, d) + dist(d, p)$$

end for

2) Tight approach for better solution: A tight approach [55] by reducing the legal capacity of each vehicle by $\theta_c\%$ and its legal working time by $\theta_t\%$ with θ_c and $\theta_t \in [0, 100)$, i.e., the right parts of (6) and (7) are replaced by $(1 - \theta_c\%)Capacity(k)$ and $(1 - \theta_t\%)WT(k)$, respectively. The reasons of implementing this on $WT(k)$ and $Capacity(k)$ are as below: *(a)* It may be wrong that the closer the current capacity or the total working time of vehicle (k) is to its limit, the better the planned routes are. *(b)* Diverse solutions can be generated by changing these two hyper-parameters. As a result, evolutionary algorithms can be used to optimise initial solutions since these solutions of high diversity could compose the population [56].

TABLE I
PROS AND CONS OF M3CWSA-I AND M3CWSA-II.

Algorithm	Pros	Cons
M3CWSA-I	1. Detailed division of all possible scenarios. 2. The distance between disposal facility and depot can be considered during the last trip.	1. Some collection sites may still not be served even if the vehicles have enough capacity to serve all the remaining sites. 2. It may be inaccurate when judging which sub-problem or combination of sub-problems currently belong to based on the relationship between $demand$ and $ability^2$. 3. It may be inaccurate when dividing the available vehicles to serve two different trips.
M3CWSA-II	Easier to understand and implement.	1. Same as Con 1 of M3CWSA-I. 2. The distance between disposal facility and depot is not considered during the last trip.

Algorithm 3 M3CWSA-II, proposed by us.

Input: A set of available vehicles V, a set of waste collection sites C, a set of disposal facilities D and a set of depots P

Apply $\mathcal{P}_1$ to all available vehicles
while the number of unserved sites > 0 **do**
    Update vehicle information and get currently available vehicles
    if no available vehicles **then**
        break
    end if
    Compute $demand$ and $ability$ for available vehicles
    Apply $\mathcal{P}_2$ to all available vehicles
end while
Add depot to each route
for any unassigned collection site c **do**
    Construct route: $p \rightarrow c \rightarrow d \rightarrow p$ with

$$p, d = \arg\min_{\forall p \in P, \forall d \in D} dist(p, c) + dist(c, d) + dist(d, p)$$

end for

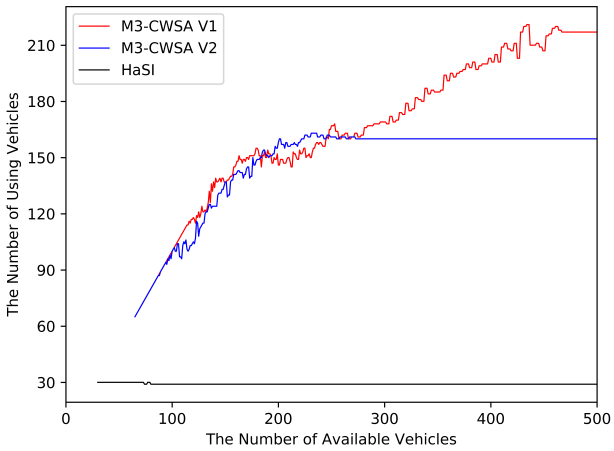
D. Experimental Study

To evaluate the effectiveness of M3CWSAs and the effectiveness of HaSI, two sets of experiments have been carried out. In the first set, we comprehensively compare HaSI and M3CWSAs (Section IV-D1). After that, we fine-tune HaSI by changing the value of θ_c and θ_t to obtain the better solutions (Section IV-D2).

All programs have been written in *C++ 11* and executed on an *Intel(R) Xeon(R) CPU E5-2650 v4* CPU working at 2.20GHz on Linux 3.10.0, using a single thread. All the experiments in this paper have been run on the same machine.

1) HaSI v.s. M3CWSAs: All available vehicles will be equally distributed to each depot in all experiments. Specifically, the extra vehicles will be allocated to each depot in turn if the number of available vehicles is not divisible by the number of depots. For example, if there are 38 available vehicles and 3 depots, the number of available vehicles in P_1, P_2 and P_3 will be 13, 13 and 12, respectively. Furthermore, the only factor which can affect the performance of M3CWSAs is the number of available vehicles, denoted as N_{AV}, since it may decide whether a waste collection site would be collected in the current trip. In other words, this waste collection site is possible to be collected in the current trip if there is at least one vehicle which still has enough capacity to collect the waste in this collection site. This possible scenario affects the performance of M3CWSAs. For HaSI, N_{AV} also has some effect on its performance due to the choice of the depot. Consequently, we test the performance of M3CWSAs and HaSI with different values of N_{AV} ($N_{AV} \in [30, 500]$). The experimental results are shown in Figure 3. In this set of experiment, the θ_c and θ_t in HaSI are all set as 1. Additionally, the speed $Speed(k)$ and maximum working time $WT(k)$ of each vehicle k are set as $45km/h$ and 8 hours, respectively.

(a) The total travel distance of feasible solutions generated by M3CWSAs and HaSI with different N_{AV}.



(b) The actual number of occupied vehicles in feasible solutions in Figure 3a.

Fig. 3. The experimental results of HaSI and M3CWSAs with different N_{AV}. Note that when the initial solution construction algorithm cannot generate feasible solution with the given N_{AV}, there is no point in illustrating the curve with that N_{AV}.

According to Figure 3, we observe that:

- The quality of the feasible solution, including total travel distance and the actual number of occupied vehicles, generated by HaSI is always higher than the one generated by M3CWSAs. Moreover, the quality of the most feasible solutions generated by M3CWSA-II is higher than the one obtained by M3CWSA-I.
- HaSI can find a feasible solution with the minimal number of actually occupied vehicles, followed by M3CWSA-I and M3CWSA- II.
- The total travel distance and the actual number of occupied vehicles in the feasible solutions generated by M3CWSAs both vibrate up and down but showing an overall upward trend as the number of available vehicles increases, and peaking at a specific number. M3CWSAs easily allocate some unserved collection sites to unoccupied vehicles rather than the vehicles which have been occupied but can still load wastes. Since adding a link between unserved site and served site needs to satisfy some stricter constraints than adding a link between two unserved sites. The number of occupied vehicles increases implies that the number of trips from and to the depots increases. Consequently, the total travel distance increases.
- The total travel distance and the actual number of occupied vehicles of the feasible solution generated by HaSI are generally steady even if they fluctuate at the beginning.
- Sometimes the total travel distance and the actual number of occupied vehicles of feasible solution decrease as the number of available vehicles increases in the local curve. The number of occupied vehicles in P_1 decreases if the number of occupied vehicles (available vehicles) in P_2 increases since there are more waste collection sites which are closed to P_2 as shown in Table II.

We highlight some results of Figure 3 in Table III. Some closer observation and conclusions which are different from the ones mentioned above can be drawn as follows: *(a)* HaSI and M3CWSAs can generate a feasible solution within a short time. *(b)* The strategy of choosing depot in HaSI still needs to be improved.In summary, HaSI can generate a better feasible solution than M3CWSAs within a shorter computational time in terms of both total travel distance and the number of occupied vehicles.

2) Fine-tuned HaSI: To get a better feasible solution, HaSI has been fine-tuned by varying the value of θ_c and θ_t where $\theta_c \in \{0, 1, \ldots, 9, 10\}$ and $\theta_t \in \{0, 1, \ldots, 19, 20\}$. In real-world applications, waste collection and transportation company should allocate the available vehicles based on the demand when the total number of the available vehicle is constant. The number of the available vehicle is set as 90 since 30 vehicles is enough for each depot.

We observe in Figure 4a that most of the feasible solutions generated by HaSI with different values of θ_c and θ_t perform well. However, there also exists some feasible solutions with poor performance. The total travel distance of the feasible solution is sensitive to the value of θ_c. The proper value for θ_c and θ_t are desirable to select in $\{1, 2, \ldots, 7\}$ and $\{0, 1, \ldots, 12\}$, respectively. The results show that making the actual load of the vehicle as close as possible to the load constraint of the vehicle will not improve the planning result.

TABLE II
HIGHLIGHTED DATA OF FEASIBLE SOLUTIONS GENERATED BY M3CWSA-II. N_V AND N_{AV} STAND FOR THE NUMBER OF OCCUPIED VEHICLES AND AVAILABLE VEHICLES, RESPECTIVELY. N_{V_i} AND N_{AV_i} WHERE $i \in \{1, 2, 3\}$ STAND FOR THE NUMBER OF OCCUPIED VEHICLES AND AVAILABLE VEHICLES IN DEPOT P_i, RESPECTIVELY. RT STANDS FOR THE RUNNING TIME OF THE ALGORITHM.

N_{AV}	Distance (m)	N_V	N_{AV_1}	N_{AV_2}	N_{AV_3}	N_{V_1}	N_{V_2}	N_{V_3}	RT (s)
100	18,606,502	100	34	33	33	34	33	33	13
101	18,512,327	101	34	34	33	34	34	33	13
102	18,673,434	102	34	34	34	34	34	34	13
103	18,471,922	100	35	34	34	32	34	34	14
104	18,656,519	100	35	35	34	31	35	34	13
105	19,221,185	104	35	35	35	34	35	35	13
106	19,212,108	104	36	35	35	34	35	35	13
107	18,410,268	97	36	36	35	26	36	35	13
108	18,533,181	97	36	36	36	25	36	36	13
109	18,410,024	96	37	36	36	24	36	36	13
110	19,176,613	103	37	37	36	30	37	36	13

TABLE III
BEST AND WORST FEASIBLE SOLUTIONS GENERATED BY HASI AND M3CWSAS. N_V AND N_{AV} STAND FOR THE NUMBER OF OCCUPIED VEHICLES AND AVAILABLE VEHICLES, RESPECTIVELY. RT STANDS FOR THE RUNNING TIME OF THE ALGORITHM.

	Distance (m)	N_{AV}	N_V	RT (s)
Best of HaSI	6,347,320	74	29	< 1
Best of M3CWSA-I	19,606,422	70	70	18
Best of M3CWSA-II	1,6059,066	81	81	17
Worst of HaSI	6,664,799	43	30	< 1
Worst of M3CWSA-I	32,441,847	$434 \sim 436$	221	13
Worst of M3CWSA-II	26,929,955	$232 \sim 233$	163	12

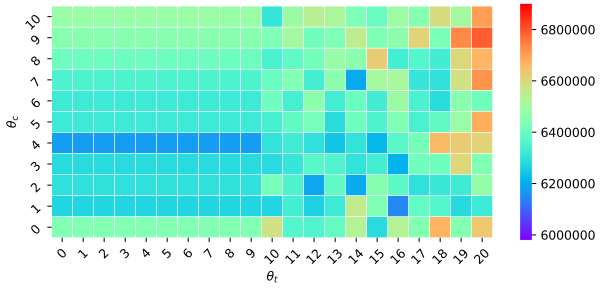
E. Discussion

It is surprising that there is a huge difference between the performance of HaSI and M3CWSAs. Some possible reasons are concluded as follows:

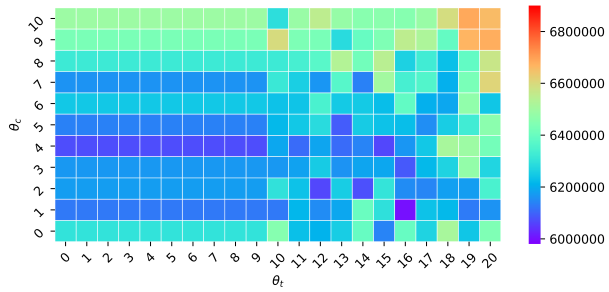
- M3CWSAs have their own obvious disadvantages, shown in Table I.
- For complex problems, likes M3CVRP, a simple solution solving strategy may be more effective. The *closest-server-first* of HaSI is simpler than the savings formula of M3CWSAs.
- The initial solution provided by CWSA without any improvement is far from the optimal one [21].
- CWSA may not be suitable to M3CVRP since not only there are more constraints, but also the distance between any two sites is asymmetrical in M3CVRP.

With the purpose of verifying the last reason, we compare the performance of CWSA and HaSI on a *Beigium* dataset[3] with single-depot, no-disposal-facility, single-trip problem instances [54]. For HaSI, θ_c is set as 0 since θ_c only has little influence on single-trip planning, and θ_t does not exist since there is no working time constraint in this dataset. The distance between any two sites is defined as the Euclidean distance. Table IV shows the experimental results. The data in column "CWSA" are directly obtained from [54]. According

[3]Available at http://vrp.atd-lab.inf.puc-rio.br/index.php/en/



(a) Total travel distance of the feasible solutions generated by HaSI with different values of θ_c and θ_t.



(b) Total travel distance of feasible solutions, in Figure 4a, optimised by BSIM.

Fig. 4. Experimental results of the feasible solutions generated by HaSI after fine-tuning (4a) and being further optimised by BSIM (4b).

to Table IV, the solutions initialised by CWSA are better than the ones by HaSI on all instances which implicitly proves our hypothesis.

TABLE IV
CWSA V.S. HASI ON *Beigium* DATASET. THE VALUES IN THE COLUMN WITH HEADER "DIFFERENCE" ARE OBTAINED BY SUBTRACTING THE VALUES IN THE "DISTANCE" COLUMN OF HASI FROM THE VALUES IN THE "DISTANCE" COLUMN OF CWSA. *RT* STANDS FOR THE RUNNING TIME OF THE ALGORITHM.

Instance (N)	CWSA		HaSI		Difference
	Distance	*RT* (s)	Distance	*RT* (s)	
L1(3,000)	200,957	6	212,601	< 1	+11,644
L2(4,000)	126,122	6	133,812	1	+7,690
A1(6,000)	497,441	12	524,027	1	+26,586
A2(7,000)	322,073	12	338,530	1	+16,457
G1(10,000)	489,556	24	507,408	2	+17,852
G2(11,000)	288,942	24	299,775	2	+10,833
B1(15,000)	531,980	36	556,691	4	+24,711
B2(16,000)	384,437	66	402,622	4	+18,185
F1(20,000)	7,518,845	102	7,719,804	8	+200,959
F2(30,000)	4,803,502	228	4,991,603	17	+188,101

V. OPTIMISATION METHODS

In this section, we first present the solution representation and evaluation function in Section V-A. Then, a novel inner-site insertion method is given in Section V-B, and the characteristics of the solution representation determine the necessity of its existence. Afterwards, two proposed optimisation methods are introduced, namely ELS (Section V-C) and Memetic Algorithm with ELS (MAELS) (Section V-D). Finally, the experimental results and discussion are shown in Section V-E and V-F, respectively.

A. Solution Representation and Evaluation

1) Representation: Our individuals are represented as permutations of the V_c tasks, without depots and disposal facilities, as in [57]. Consequently, the depots and disposal facilities in the solutions generated by HaSI need to be deleted. When evaluating an individual, the depots and disposal facilities are inserted for calculating (3). Therefore, an insertion method (detailed in Section V-B), which does not make the solution after inserting depot and disposal facilities worse than the initial one before deleting the depots and disposal facilities, is needed.

2) Evaluation function: Same as in [26], the function for evaluating solutions is defined as

$$f(S) = Cost(S) + \gamma V(S), \tag{13}$$

where $Cost(S)$ is the total travel distance of S calculated with (3) and $V(S)$ is the total violation of S, calculated as the summation of the violations of all routes in S. The coefficient γ is calculated as $\frac{Cost(S_{best})}{\mu}(\frac{Cost(S_{best})}{Cost(S)} + \frac{Cost(S)}{\mu} + 1)$, where the best-so-far solution is represented as S_{best} and the population size is μ . More details can be found in [26]. The population size μ is set as 1 in ELS.

B. Backspacing Insertion Method

In a real-life situation, it may be optimal for a vehicle to dump in advance rather than keeping collecting until the upper bound of load constraint is met. The backspacing insertion method (BSIM) is proposed on account of such kind of situation.

The key of BSIM is described as follows. Firstly, given a route R_k for a vehicle k, BSIM will find the latest collection site (indexed by τ) in the first trip that a vehicle can serve before being fulfilled. Afterwards, BSIM will loop over and record all the cases of inserting the closest disposal facility from $(\tau - m)^{th}$ to τ^{th} collection site, where $m \in \{0, 1, \ldots, N_b\}$. The next step will start from the latest collection site of next trip and repeat the preceding two steps for generating each trip in route R_k. When BSIM reaches the last trip of route R_k, the distance $D_d = dist(d, c_{k,1,j}) + dist(c_{k,l_k,h_{k,i}}, df) + dist(df, d)$ is calculated for each $d \in V_P$, where df represents the last disposal facility in R_k. Then, the depot with the shortest D_d is chosen to be the depot of R_k. Finally, the route R_k's total distances of all possible combination of cases from the record are calculated, and the best disposal facility insertion plan is selected as the final result.

C. Extended Local Search

By designing 3 novel search operators that are more suitable for M3CVRP, we have extended the local search algorithm as shown in Algorithm 4. ρ_1, ρ_2, ρ_3 and α are control parameters.

1) Region-constrained single-point swap (RCSPS): For each collection site c in a randomly selected route R_k from a given solution S, if there exists at least one site in other routes that locate within a certain distance ρ_1, then a site is randomly picked up. Let c' denote the selected site, and

$R_{k'}$ is its corresponding route. Two new routes, $R_k^{'}$ and $R_{k'}^{'}$, can be obtained after switching c and c'. If the inequality $C_r(R_k^{'})+C_r(R_{k'}^{'}) < C_r(R_k)+C_r(R_{k'})$ holds, then this swap operation will be accepted; otherwise, nothing will happen.

2) Region-constrained segment swap (RCSS): For a given solution S, a waste collection site c is randomly selected from a route R_k selected uniformly at random. If there exists at least one site in other routes that locate within a certain distance ρ_2, then randomly choose one c' from route $R_{k'}$ among all alternatives. The two new routes, $R_k^{'}$ and $R_{k'}^{'}$, can be obtained after swapping the whole segments after c and c'. If $C_r(R_k^{'})+C_r(R_{k'}^{'}) < C_r(R_k)+C_r(R_{k'})$, then the swap operation is accepted; otherwise, nothing will happen.

3) Relaxed multi-point swap (RMPS): For a given solution S, two routes R_1 and R_2 are randomly selected. A small proportion are randomly selected from each route, denoted as B_1 and B_2. Two new routes R_1' and R_2' are generated by randomly inserting the sites of B_2 into R_1 and the sites of B_1 into R_2 with the best position, respectively. The best position is the one where the lowest cost (calculated with (3)) is achieved, and the insertion order of sites in B_1 and B_2 is random. If $C_r(R_1')+C_r(R_2') \leq C_r(R_1)+C_r(R_2)+\rho_3$, R_1 and R_2 are replaced by R_1' and R_2', respectively. $C_r(R)$ is the total travel distance of a route R, detailed previously in Section III-B.

4) Insertion: Arbitrarily select a waste collection site c from a randomly selected route R_k in a given solution S, and then re-insert c into the best position of R_k to obtain a new route $R_k^{'}$. The definition of the best position is the same as the best position mentioned in RMPS. If $C_r(R_k^{'}) < C_r(R_k)$, then the insertion operation is accepted; otherwise, no operation occurs.

5) Swap: In the swap search operator, two waste collection sites are randomly selected from a randomly selected route R_k, and then their positions are exchanged to obtain a new route $R_k^{'}$. Similarly, if $C_r(R_k^{'}) < C_r(R_k)$, then the swap operation is accepted; otherwise, no operation occurs.

6) 2-opt: Two kinds of 2-opt search operators exist, one for a single route and the other for double routes [26]. Only the operator for a single route is employed in ELS since region-constrained segment swap has similar search operation as the 2-opt for double routes. In the 2-opt operator for a single route, two waste collection sites c_1 and c_2 are randomly selected from a randomly selected route R_k, and then a new route $R_k^{'}$ is obtained after revering the visiting order of the waste collection sites between c_1 and c_2. Similarly, if $C_r(R_k^{'}) < C_r(R_k)$, then this type of reverse operation is accepted; otherwise, no operation occurs.

Our extended local search (ELS) applies the above novel search operators together with three classic search operators, *insertion*, *swap* and *2-opt* [35], in an order defined in Algorithm 4, to improve input solutions.

The main idea of designing RCSPS and RCSS are as follows: *(a)* Reducing search space with its "region-constrained" since the search space for large-scale M3CVRP is huge. *(b)* Increasing search step size for being capable of searching within a broader neighbourhood since those three classic search operators have small search step size such that they can only search within a small neighbourhood [26]. The purpose of RMPS is to help other search operators jump out of the local optimum with its "relaxation" of ρ_3.

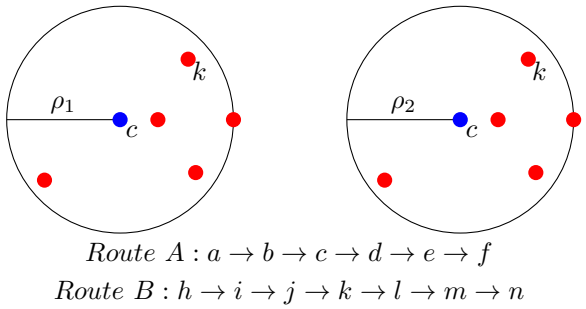


Fig. 5. Demonstration of search operators RCSPS (*left*) and RCSS (*right*), assuming two routes A and B. If the swap condition is valid, RCSPS will replace A and B will by $a \rightarrow b \rightarrow k \rightarrow d \rightarrow e \rightarrow f$ and $h \rightarrow i \rightarrow j \rightarrow c \rightarrow l \rightarrow m \rightarrow n$, respectively, while RCSS will change A and B to $a \rightarrow b \rightarrow k \rightarrow l \rightarrow m \rightarrow n$ and $h \rightarrow i \rightarrow j \rightarrow c \rightarrow d \rightarrow e \rightarrow f$, respectively.

Algorithm 4 ELS, proposed by us in Section V-C. Stopping criteria SC_1 and SC_2 are user-defined.

Input: A original solution S; control parameters ρ_1, ρ_2, ρ_3 and α

```
S' ← S
while stopping criterion SC1 not met do
    while stopping criterion SC2 not met do
        Apply RCSPS with parameter ρ1 on S'
        Apply RCSS with parameter ρ2 on S'
        Apply swap, insertion and 2-opt operators each once in a random order on S'
    end while
    Apply RMPS with parameters ρ3 and α on S'
end while
return S'
```

D. Memetic Algorithm with ELS (MAELS)

The Memetic Algorithm (MA) was introduced by Moscato in 1989 [23]. Comparing to GA, the mutation operators of GA are replaced by local search in MA, which is the main difference between MA and GA [26]. Consequently, MA can balance well between generality and problem specificity due to the domain-specific heuristics embedded in local search operators [26]. In this paper, we integrate ELS to an MA [23], [26]. The details of our MA with ELS (MAELS) are given in Algorithm 5. It is important for MA to have an initial population with high diversity. The difference among the solutions in a population is the set of collection sites allocated to the same vehicle and the traversal order among these sites. The trip is the smallest set of sites in a population. Moreover, we do not consider the traversal order of sites in trips since exact methods can found the optimal order. Consequently, we measure the diversity of the initialised population [58] using the an entropy defined as

$$H = -\sum_{trip \in T} \Gamma_{trip} \log \Gamma_{trip}, \tag{14}$$

where $\Gamma_{trip} = \frac{n_{trip}}{\mu}$ and $n_{trip} = \sum_{i=1}^{\mu} \mathbf{1}(trip \in T_i)$. T_i denotes the set of trips appeared in solution S_i, $\forall i \in \{1, \ldots, \mu\}$. T is the set of all trips appeared in the population, i.e., $T = \cup_{i=1}^{\mu} T_i$. For all $i \in \{1, \ldots, \mu\}$, $\mathbf{1}(trip \in T_i) = 1$ if $trip \in T_i$ is true; otherwise, $\mathbf{1}(trip \in T_i) = 0$.

Algorithm 5 MAELS, proposed by us in Section V-D. SBX refers to a sequence-based crossover operator [59].

Input: an initial population P of size μ, a maximum number of offspring λ, and p_{ELS}, the probability of applying ELS

while time not elapsed **do**
  offspring$\leftarrow \emptyset$
  for $i = 1, 2, \ldots, \lambda$ **do**
    Randomly select two distinct solutions S_1 and S_2 from P as parents
    Apply the SBX operator to S_1 and S_2 to generate two new solutions, then evaluate and keep the one with shorter distance as S'
    if $UniformRandom(0, 1) < p_{ELS}$ **then**
      $S'' \leftarrow$ Apply ELS to S'
      if S'' is not a clone of any solution in offspring or P **then**
        Add S'' to offspring
      else if S' is not a clone of any solution in offspring or P **then**
        Add S' to offspring
      end if
    else if S' is not a clone of any solution in offspring or P **then**
      Add S' to offspring
    end if
  end for
  Sort the solutions in $P \cup$ offspring using stochastic ranking [60]
  Set P = {the best μ individuals in $P \cup$ offspring}
end while
return the best feasible solution in P

E. Experimental Study

With the intention of better evaluating the BSIM, novel search operators, ELS and MAELS, three sets of experiments have been carried out. In the first set, we use BSIM to insert disposal facilities and depots into solutions, i.e., permutations of waste collection sites (Section V-E1). After that, we comprehensively compare the effectiveness and contribution of each novel operator in ELS in Section V-E2. A comparison of optimisation effect between ELS and MAELS is performed on the last set (Section V-E3). Besides, we preliminary verify the validity of the diversity measure described in Section V-E3.

1) Effectiveness of BSIM: To verify the effectiveness of BSIM, we optimised the feasible solutions generated by HaSI, the quality of which are demonstrated in Figure 4a. First, the depots and disposal facilities in those solutions are deleted. Then, BSIM inserts the depots and disposal facilities in its way. The optimisation results are shown in Figure 4b with the control parameter N_b=5. By comparing Figures 4b and 4a, we can conclude that BSIM can optimise the feasible solutions generated with $\theta_c \in \{0, 1, \ldots, 8\}$ and cannot optimise part of the feasible solutions generated with $\theta_c \in \{9, 10\}$. The reason is that the load of vehicle k is impossible to reach the level of $0.9 * Capacity(k)$ or $0.91 * Capacity(k)$ when N_b is set as 5.

In conclusion, BSIM has the probability to optimise the solutions and will not make solutions worse by re-inserting depots and disposal facilities as long as N_b is large enough.

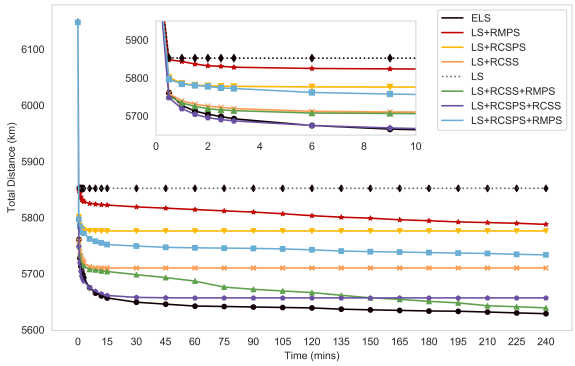


Fig. 6. Comparison between LS, LS with all combinations of novel search operators and ELS in terms of total travel distance. Table V presents the corresponding figures and standard deviations.

2) LS v.s. ELS: In order to better understand the effectiveness and contribution of each novel search operator in ELS, a comparative experiment has been carried out, and the experimental results are shown in Figure 6 and Table V. We compare LS, ELS and different usage combination of the three novel search operators (RMPS, RCSS and RCSPS) to exploit the impact of each operator. The control parameters ρ_1 and ρ_2 are set as $5km$, and ρ_3 is set as $15km$. α is set as $1/|R_k|$ where R_k is randomly picked in RMPS. In other words, only one point will be swapped in RMPS. Additionally, the stopping criteria SC_1 and SC_2 are if the running time of the program has been exceeded 4 hours and if the reduction of total travel distance does not exceed $2km$ for 5 consecutive iterations, respectively. This parameter setting is arbitrarily chosen.

Figure 6 and Table V show that:

(i) RMPS can help other search operators to jump out of the local optimum, but its step size is relatively small at each generation. Only one point will be swapped in RMPS, which rarely leads to large mutations. RMPS relaxes too much would result in skipping the neighbourhood of global optimum. Determining the optimal degree of relaxation for efficient convergence is crucial.

(ii) Both RCSS and RCSPS can perform efficient searches, but premature convergence occurs. It may be explained by the "region-constrained" feature which limits the search space of RCSPS and RCSS and the right search direction driven by heuristics.

(iii) Within the first few minutes or less, the average distances of solutions optimised by LS variants embedded with

TABLE V

THE COMPARISON BETWEEN ELS, LS AND LS WITH DIFFERENT SINGLE NOVEL SEARCH OPERATOR. "BEST" AND "AVG" STAND FOR THE BEST AND AVERAGE DISTANCES OBTAINED FROM 30 INDEPENDENT RUNS. "STD" STANDS FOR THE STANDARD DEVIATION. THE LOWEST AVERAGE DISTANCE IS MARKED WITH *. RESULTS IN BOLD (OR UNDERLINE) INDICATE THAT THE CORRESPONDING ALGORITHM IS BETTER (OR WORSE) THAN LS BASED ON WILCOXON RANK-SUM TEST WITH THE LEVEL OF SIGNIFICANCE 0.05.

Time	LS			LS+RCSPS			LS+RCSS			LS+RMPS			LS+RCSPS+RMPS			LS+RCSS+RMPS			LS+RCSPS+RCSS			ELS		
(mins)	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD
0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0
0.5	5844.9	5853.1	4.7	5781.5	**5801.6**	14.9	5692.4	**5758.8**	20.4	5819.6	5848.2	12.7	5776.7	**5796.9**	11.5	5715.3	**5752.8**	16.1	5719.0	**5748.6***	16.2	5719.2	**5761.3**	20.7
1	5843.9	5852.4	4.6	5771.7	**5786.3**	9.8	5664.5	**5740.0**	25.8	5815.5	**5843.4**	13.4	5767.6	**5784.6**	8.0	5694.1	**5734.5**	18.3	5688.7	**5719.5***	14.5	5695.5	**5727.7**	15.6
1.5	5843.2	5852.3	4.7	5769.5	**5781.7**	7.7	5651.4	**5731.3**	26.2	5809.2	**5837.1**	16.1	5743.5	**5780.5**	10.2	5683.1	**5725.6**	20.2	5674.4	**5704.4***	13.6	5686.7	**5713.2**	16.8
2	5843.2	5852.3	4.7	5768.7	**5779.9**	7.7	5650.6	**5726.0**	26.6	5808.6	**5832.1**	15.4	5739.0	**5777.9**	10.6	5681.0	**5719.3**	21.6	5668.0	**5695.6***	14.5	5680.8	**5704.8**	17.3
2.5	5843.2	5852.3	4.7	5767.7	**5778.8**	7.4	5650.6	**5722.9**	26.4	5808.0	**5831.0**	14.7	5737.4	**5774.1**	13.0	5680.4	**5716.0**	22.1	5662.1	**5690.7***	14.4	5673.1	**5698.6**	17.4
3	5843.2	5852.3	4.7	5766.1	**5778.2**	7.6	5648.6	**5719.6**	26.3	5808.0	**5828.4**	12.8	5733.2	**5772.6**	13.8	5674.0	**5714.2**	23.4	5660.5	**5687.2***	15.0	5668.7	**5692.8**	16.9
6	5843.2	5852.3	4.7	5764.6	**5776.7**	7.7	5646.8	**5712.5**	27.4	5803.0	**5825.2**	13.1	5732.9	**5762.1**	16.8	5645.8	**5707.9**	26.5	5631.7	**5675.3**	16.0	5641.1	**5675.1***	15.7
9	5843.2	5852.3	4.7	5764.6	**5776.4**	7.7	5646.8	**5710.8**	27.5	5803.0	**5824.1**	13.1	5732.6	**5757.9**	16.2	5644.8	**5706.9**	25.8	5619.5	**5668.7**	17.0	5618.4	**5665.7***	19.2
12	5843.2	5852.3	4.7	5764.4	**5776.3**	7.7	5646.8	**5710.4**	27.8	5801.6	**5823.0**	12.5	5731.1	**5754.9**	14.1	5644.8	**5705.2**	26.4	5619.5	**5664.0**	17.9	5613.1	**5660.7***	19.0
15	5843.2	5852.3	4.7	5764.4	**5776.3**	7.8	5646.8	**5710.4**	27.8	5801.6	**5822.4**	12.5	5731.1	**5752.0**	12.5	5644.8	**5704.0**	26.8	5619.4	**5661.3**	19.2	5605.7	**5656.9***	19.9
30	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5801.6	**5819.1**	12.7	5724.5	**5749.3**	11.8	5644.8	**5698.5**	26.9	5618.2	**5657.9**	20.0	5601.7	**5649.3***	20.6
45	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5798.9	**5816.9**	12.7	5724.5	**5747.1**	10.4	5643.8	**5693.2**	28.5	5618.2	**5657.2**	20.0	5601.5	**5645.9***	21.3
60	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5796.1	**5814.4**	12.6	5724.5	**5746.3**	9.7	5624.8	**5686.9**	30.3	5618.2	**5656.9**	20.1	5601.5	**5642.4***	21.8
75	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5796.1	**5812.1**	11.5	5724.5	**5745.6**	9.8	5624.4	**5676.2**	26.7	5618.2	**5656.9**	20.1	5601.5	**5641.7***	22.0
90	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5794.2	**5809.9**	11.7	5722.7	**5745.2**	9.9	5624.4	**5672.3**	26.1	5618.2	**5656.9**	20.1	5601.5	**5640.8***	22.6
105	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5783.1	**5806.9**	11.5	5722.7	**5744.2**	9.2	5613.2	**5669.3**	26.5	5618.2	**5656.9**	20.1	5601.5	**5640.0***	22.8
120	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5783.1	**5803.5**	9.4	5722.7	**5742.6**	8.8	5610.1	**5666.5**	26.9	5618.2	**5656.9**	20.1	5599.3	**5639.1***	22.5
135	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5775.1	**5800.8**	10.8	5719.0	**5740.5**	9.6	5569.3	**5661.6**	30.4	5618.2	**5656.9**	20.1	5597.2	**5636.9***	20.5
150	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5775.1	**5799.0**	11.2	5712.7	**5739.4**	10.1	5565.9	**5657.1**	28.9	5618.2	**5656.9**	20.1	5583.4	**5635.7***	21.8
165	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5774.1	**5796.2**	11.6	5711.2	**5738.7**	10.1	5563.9	**5654.4**	29.3	5618.2	**5656.9**	20.1	5569.2	**5634.7***	23.4
180	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5773.2	**5794.5**	12.1	5711.2	**5737.7**	9.6	5563.1	**5650.8**	30.4	5618.2	**5656.9**	20.1	5568.5	**5633.7***	23.8
195	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5772.9	**5792.3**	12.5	5711.2	**5736.8**	10.0	5561.2	**5648.1**	30.5	5618.2	**5656.9**	20.1	5568.5	**5633.2***	24.0
210	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5772.9	**5791.2**	12.5	5711.2	**5736.2**	10.1	5561.0	**5643.0**	31.7	5618.2	**5656.9**	20.1	5568.5	**5631.8***	24.2
225	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5772.9	**5790.0**	12.8	5693.8	**5734.7**	12.5	5561.0	**5641.2**	31.5	5618.2	**5656.9**	20.1	5568.5	**5630.3***	24.2
240	5843.2	5852.3	4.7	5764.4	**5776.2**	7.7	5646.8	**5710.4**	27.8	5763.6	**5788.0**	13.8	5693.8	**5733.7**	12.7	5561.0	**5639.2**	30.8	5618.2	**5656.9**	20.1	5568.5	**5628.8***	22.9

RCSS are slightly lower than the ones by ELS (cf. Table V). Since RCSS in ELS has iterated fewer times than LS variants embedded with RCSS within the same time.

(iv) RCSPS and RCSS have the steadiest and the most unstable optimisation effect among three novel search operators, respectively. For RCSPS, the waste collection site can only be swapped with the ones within a certain distance ρ_1. Therefore, the solutions of independent optimisation runs of RCSPS are similar due to the small search space of RCSPS. On the contrary, the optimised solutions by RCSS are quite different since RCSS's search space is larger than the one of RCSPS. The benefit from RCSPS or RCSS in terms of reduced distance by RCSPS or RCSS is limited, but these two operators significantly change the neighbourhood that will be searched by insert, 2-Opt and swap operators.

In summary, RMPS mainly helps ELS jump out of the local optimum, and RCSPS not only makes ELS have steady optimisation performance but also help ELS reach a better solution more efficiently. For ELS, on the one hand, RCSS can change a lot the neighbourhood which is then searched by other operators. On the other hand, RCSS can improve the efficiency of ELS.

3) ELS v.s. MAELS: Three questions are investigated in this subsection: *(RQ1)* Can the MA framework help ELS jump out of the local optimum? *(RQ2)* How much does the MA contribute to the MAELS framework? *(RQ3)* Is the proposed population diversity measure (14) valid?

For parameter setting, the population size μ of MAELS/MALS is 30, and the size of offspring λ and the probability of applying ELS/LS in MAELS/MALS p_{ELS} are set as 6μ and 0.2, respectively. All these parameters are set to the same as the parameters in [26]. The Figure 4a illustrates that the initial population of MAELS/MALS with high (or low) diversity is chosen from the population with highest (or lowest) diversity among $10,000$ populations which are obtained by randomly choosing 30 feasible solutions from the set of solutions. To make the comparison between ELS and MAELS more comprehensible, the initial population must contain the feasible solution optimised by ELS in the last set of experiment. In other words, there are only 29 feasible solutions being picked randomly in the initial solutions. Expressly, stopping criteria SC_1 of ELS used in MAELS is set as if the number of iterations of that loop has been exceeded 5. The SC_1 and SC_2 are arbitrarily set.

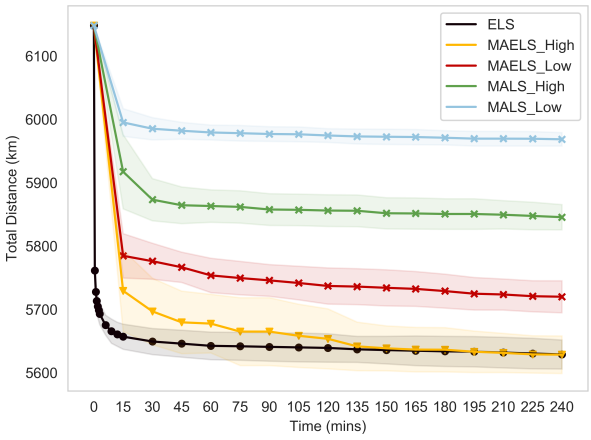
The experimental results (Table VI and Figure 7a) illustrate that:

- MAELS has better optimisation performance than MALS.
- MAELS/MALS has better optimisation effect to the high diversity initial population than to the low diversity initial population.
- ELS can obtain better solutions than MAELS_High (i.e., MAELS initialised with a population of high diversity) during the early stage of optimisation. Though, MAELS_High can get solutions of similar or even better quality than ELS after the optimisation time increased to 135 minutes.

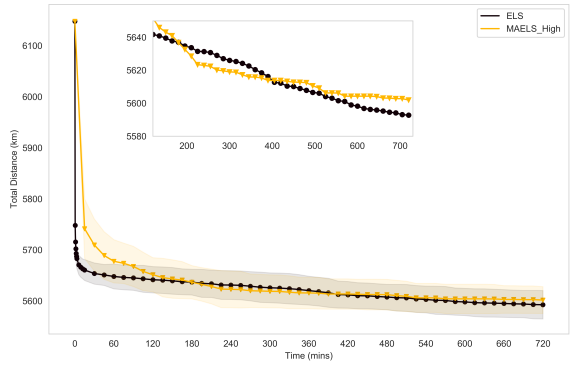
In the solutions optimised by MAELS_High among 30 independent runs, there are few vehicles (the maximum number is 3) whose working time exceeded 8 hours and up to 20 minutes in some solutions. It is usually acceptable in real applications. Additionally, with a focus on studying the optimisation effect of ELS and MAELS after optimising for a longer time, we perform a supplementary experiment and illustrate the results

in Figure 7b. After three hours' optimisation, there is no significant difference between the optimisation effects of ELS and MAELS.

In conclusion, our novel diversity measurement is sufficient. Moreover, ELS plays an essential role in the optimisation process of MAELS. Although the framework of MA might helps the ELS to jump out of the local optimum, MAELS needs to take more time on optimising.



(a) MAELS and MALS are initialised with two identical populations of low and high diversity, separately.



(b) ELS and MAELS with longer running time.

Fig. 7. Total travel cost of ELS, compared with various local search algorithms. The shaded parts indicate the standard deviations.

F. Discussion

MAELS cannot guarantee that the optimised solutions do not exceed the maximum working time of vehicles. Nevertheless, a small amount of extra working time is acceptable as the current model does not take into account the uncertainties in real life, which may lead to longer or shorter service time. Additionally, including stricter constraints, such as refusing to add infeasible individuals to the population, is possible but not necessary. Since it may reduce the ability of MAELS to jump out of the local optimal, thereby making its optimisation effect worse. On the other hand, relaxing the constraints of the maximum working time of vehicles (e.g., changing from 8 hours to 8.5 hours) in ELS/MAELS may increase the probability of rejecting better solutions. Some favourable solutions may need to be obtained by the optimisation on infeasible initial solutions. MAELS may speed up its optimisation if the best individual in the population is always selected as one of the parents for crossover.

VI. Conclusion

In this paper, we formulate a multi-depot multi-disposal-facility multi-trip capacitated vehicle routing problem (M3CVRP) for a real-world large-scale waste collection problem with many constraints that no existing work has ever solved such problem of this size. We design a heuristic-assisted solution initialisation approach (HaSI) for generating feasible initial solutions of high quality, combining the idea of greedy strategy with the consideration of tight capacity and working time. Due to the characteristics of solution representation during optimisation, a better insertion method, backspacing insertion method (BSIM), is proposed. Moreover, two novel search operators considering regional information and one novel search operators helping to jump out of the local optimum have been designed and embedded in an extended local search (ELS) algorithm, as well as a memetic algorithm with ELS (MAELS) to optimise feasible initial solutions further. The HaSI is compared with the adaptations of Clarke and Wright Savings Algorithm. Experimental study shows that HaSI is capable of generating high-quality and diverse initial solutions. By re-allocating the depots and disposal facilities on these initial solutions, BSIM significantly reduces most of the solutions. Compared with MALS, a sophisticated memetic algorithm, ELS can efficiently find good solutions. In particular, ELS can rapidly find high-quality solutions in a very short period, which makes it suitable for real-life applications. For some incredibly complex problems, a greedy strategy can help to find feasible solutions within a very short time though it may not find the optimal solution. Additionally, some additional traversals within a short time may help to improve the solution obtained by the greedy strategy, like BSIM proposed in this paper.

There are several potential research directions as future work. Designing a search operator or heuristic to adapt the number of vehicles in the optimisation process needs to be researched. In order to speed up the optimisation of MAELS, MAELS can include heuristics to apply ELS to only the best feasible solution or those "promising" (e.g., with low cost and feasible) solutions [26].

References

[1] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management Science*, vol. 6, no. 1, pp. 80–91, 1959.

[2] J. K. Lenstra and A. R. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.

[3] B. Eksioglu, A. V. Vural, and A. Reisman, "The vehicle routing problem: A taxonomic review," *Computers & Industrial Engineering*, vol. 57, no. 4, pp. 1472–1483, 2009.

[4] R. Baldacci, A. Mingozzi, and R. Roberti, "Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints," *European Journal of Operational Research*, vol. 218, no. 1, pp. 1–6, 2012.

TABLE VI
THE COMPARISON BETWEEN ELS, MALS AND MAELS INITIALISED WITH POPULATION OF DIFFERENT DIVERSITY DEGREE. "BEST" AND "AVG" STAND FOR THE BEST AND AVERAGE DISTANCES OBTAINED FROM 30 INDEPENDENT RUNS. "STD" STANDS FOR THE STANDARD DEVIATION. THE LOWEST AVERAGE DISTANCE IS MARKED WITH *. RESULTS IN BOLD (OR UNDERLINE) INDICATE THAT THE CORRESPONDING ALGORITHM IS BETTER (OR WORSE) THAN ELS BASED ON WILCOXON RANK-SUM TEST WITH THE LEVEL OF SIGNIFICANCE 0.05.

Time	ELS			MAELS_High			MAELS_Low			MALS_High			MALS_Low		
(mins)	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD	BEST	AVG	STD
0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0	6148.0	6148.0	0.0
15	5605.7	5656.9*	19.9	5574.7	5729.3	64.4	5704.2	5784.5	35.1	5835.5	5917.4	58.6	5943.2	5995.1	521.9
30	5601.7	5649.3*	20.6	5574.7	5696.8	52.5	5704.2	5776.0	28.4	5835.5	5873.2	33.3	5943.2	5985.2	17.6
45	5601.5	5645.9*	21.3	5574.7	5679.5	49.4	5704.2	5766.5	23.9	5817.9	5864.4	29.0	5943.2	5982.0	13.3
60	5601.5	5642.4*	21.8	5574.7	5677.5	46.3	5660.8	5753.6	27.1	5817.9	5863.1	25.6	5943.2	5979.3	11.8
75	5601.5	5641.7*	22.0	5505.4	5664.9	53.6	5660.8	5749.4	26.4	5817.9	5861.8	25.6	5943.2	5978.1	12.3
90	5601.5	5640.8*	22.6	5505.4	5664.9	53.6	5660.8	5745.7	25.0	5817.9	5857.6	25.1	5943.2	5976.9	11.6
105	5601.5	5634.0*	22.8	5505.4	5658.3	49.9	5660.8	5741.6	26.6	5817.9	5857.1	24.9	5943.2	5976.4	11.7
120	5599.3	5639.1*	22.5	5505.4	5653.5	47.4	5660.8	5736.9	28.6	5817.6	5856.0	25.1	5943.2	5974.5	10.8
135	5597.2	5636.9*	20.5	5505.4	5641.6	38.3	5660.8	5734.0	28.4	5817.6	5855.7	25.1	5943.2	5973.0	11.8
150	5583.4	5635.7*	21.8	5505.4	5638.5	35.5	5660.8	5734.0	28.6	5812.6	5851.7	24.6	5943.2	5972.5	12.0
165	5569.2	5634.7*	23.4	5505.4	5636.5	35.1	5660.8	5732.3	27.5	5812.6	5851.4	24.7	5943.2	5972.0	12.0
180	5568.5	5633.7*	23.8	5505.4	5636.3	35.0	5660.8	5728.9	26.8	5812.6	5850.6	24.2	5943.2	5970.9	11.2
195	5568.5	5633.2*	24.0	5505.4	5633.1	32.8	5660.8	5724.7	25.7	5812.6	5850.6	24.2	5943.2	5969.5	11.6
210	5568.5	5631.8	24.2	5505.4	5631.3*	30.9	5660.8	5723.5	24.6	5812.6	5849.4	22.3	5943.2	5969.5	11.6
225	5568.5	5630.3	24.2	5505.4	5628.9*	29.5	5660.8	5720.8	24.5	5812.6	5847.5	21.8	5943.2	5969.3	11.4
240	5568.5	5628.8	22.9	5505.4	5628.5*	29.7	5660.8	5719.9	25.3	5812.6	5845.5	19.9	5943.2	5968.5	10.6

[5] R. Liu, Z. Jiang, R. Y. Fung, F. Chen, and X. Liu, "Two-phase heuristic algorithms for full truckloads multi-depot capacitated vehicle routing problem in carrier collaboration," *Computers & Operations Research*, vol. 37, no. 5, pp. 950–959, 2010.

[6] A. Mingozzi, R. Roberti, and P. Toth, "An exact algorithm for the multitrip vehicle routing problem," *INFORMS Journal on Computing*, vol. 25, no. 2, pp. 193–207, 2013.

[7] S. Ubeda, F. J. Arcelus, and J. Faulin, "Green logistics at eroski: A case study," *International Journal of Production Economics*, vol. 131, no. 1, pp. 44–51, 2011.

[8] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 1, pp. 70–85, 2016.

[9] B.-I. Kim, S. Kim, and S. Sahoo, "Waste collection vehicle routing problem with time windows," *Computers & Operations Research*, vol. 33, no. 12, pp. 3624–3642, 2006.

[10] E. J. Beltrami and L. D. Bodin, "Networks and vehicle routing for municipal waste collection," *Networks*, vol. 4, no. 1, pp. 65–94, 1974.

[11] A. Gruler, C. Fikar, A. A. Juan, P. Hirsch, and C. Contreras-Bolton, "Supporting multi-depot and stochastic waste collection management in clustered urban areas via simulation–optimization," *Journal of Simulation*, vol. 11, no. 1, pp. 11–19, 2017.

[12] J. Liu and Y. He, "A clustering-based multiple ant colony system for the waste collection vehicle routing problems," *Energy Procedia*, vol. 11, no. 1, pp. 3397–3405, 2012.

[13] J. Beliën, L. De Boeck, and J. Van Ackere, "Municipal solid waste collection and management problems: a literature review," *Transportation Science*, vol. 48, no. 1, pp. 78–102, 2014.

[14] S. Martello, D. Pisinger, and D. Vigo, "The three-dimensional bin packing problem," *Operations Research*, vol. 48, no. 2, pp. 256–267, 2000.

[15] S. Martello, D. Pisinger, D. Vigo, E. D. Boef, and J. Korst, "Algorithm 864: General and robot-packable variants of the three-dimensional bin packing problem," *ACM Transactions on Mathematical Software (TOMS)*, vol. 33, no. 1, pp. 7–es, 2007.

[16] M. Akhtar, M. Hannan, R. A. Begum, H. Basri, and E. Scavino, "Backtracking search algorithm in cvrp models for efficient solid waste collection and route optimization," *Waste Management*, vol. 61, pp. 117–128, 2017.

[17] É. D. Taillard, "A heuristic column generation method for the heterogeneous fleet vrp," *RAIRO-Operations Research-Recherche Opérationnelle*, vol. 33, no. 1, pp. 1–14, 1999.

[18] K. C. Tan, L. H. Lee, Q. Zhu, and K. Ou, "Heuristic methods for vehicle routing problem with time windows," *Artificial intelligence in Engineering*, vol. 15, no. 3, pp. 281–295, 2001.

[19] G. Laporte and F. Semet, "Classical heuristics for the capacitated vrp," in *The vehicle routing problem*. SIAM, 2002, pp. 109–128.

[20] G. Clarke and J. W. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Operations Research*, vol. 12, no. 4, pp. 568–581, 1964.

[21] T. Pichpibul and R. Kawtummachai, "An improved clarke and wright savings algorithm for the capacitated vehicle routing problem," *Sciencea-sia*, vol. 38, no. 3, p. 307, 2012.

[22] M. Misic, A. Dordevic, and A. K. Arsic, "The optimization of vehicle routing of communal waste in an urban environment using a nearest neighbirs' algorithm and genetic algorithm: Communal waste vehicle routing optimization in urban areas," in *2017 Ninth International Conference on Advanced Computational Intelligence (ICACI)*, 2017, pp. 264–271.

[23] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," *Caltech Concurrent Computation Program, C3P Report*, vol. 826, p. 1989, 1989.

[24] T. T. Minh, T. Van Hoai, and T. T. N. Nguyet, "A memetic algorithm for waste collection vehicle routing problem with time windows and conflicts," in *International Conference on Computational Science and Its Applications*. Springer, 2013, pp. 485–499.

[25] A. M. Benjamin and J. Beasley, "Metaheuristics with disposal facility positioning for the waste collection vrp with time windows," *Optimization Letters*, vol. 7, no. 7, pp. 1433–1449, 2013.

[26] K. Tang, Y. Mei, and X. Yao, "Memetic algorithm with extended neighborhood search for capacitated arc routing problems," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1151–1166, 2009.

[27] Y. Mei, X. Li, and X. Yao, "Decomposing large-scale capacitated arc routing problems using a random route grouping method," in *2013 IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 1013–1020.

[28] Y. Mei, X. Li, and X. Yao, "Cooperative coevolution with route distance grouping for large-scale capacitated arc routing problems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 435–449, 2014.

[29] K. Tang, J. Wang, X. Li, and X. Yao, "A scalable approach to capacitated arc routing problems based on hierarchical decomposition," *IEEE Transactions on Cybernetics*, vol. 47, no. 11, pp. 3928–3940, 2017.

[30] S. Lin, "Computer solutions of the traveling salesman problem," *Bell System Technical Journal*, vol. 44, no. 10, pp. 2245–2269, 1965.

[31] H. Handa, D. Lin, L. Chapman, and Xin Yao, "Robust solution of salting route optimisation using evolutionary algorithms," in *2006 IEEE International Conference on Evolutionary Computation*, 2006, pp. 3098–3105.

[32] H. Handa, L. Chapman, and X. Yao, "Robust route optimization for gritting/salting trucks: a cercia experience," *IEEE Computational Intelligence Magazine*, vol. 1, no. 1, pp. 6–9, 2006.

[33] A. Hertz, G. Laporte, and M. Mittaz, "A tabu search heuristic for the capacitated arc routing problem," *Operations Research*, vol. 48, no. 1, pp. 129–135, 2000.

[34] A. Hertz and M. Mittaz, "A variable neighborhood descent algorithm for the undirected capacitated arc routing problem," *Transportation Science*, vol. 35, no. 4, pp. 425–434, 2001.

[35] P. Beullens, L. Muyldermans, D. Cattrysse, and D. V. Oudheusden, "A guided local search heuristic for the capacitated arc routing problem," *European Journal of Operational Research*, vol. 147, no. 3, pp. 629–643, 2003.

[36] J. Brandao and R. W. Eglese, "A deterministic tabu search algorithm for the capacitated arc routing problem," *Computers & Operations Research*, vol. 35, no. 4, pp. 1112–1126, 2008.

[37] L. Song and Y. Dong, "An improved differential evolution algorithm with local search for capacitated vehicle routing problem," in *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, 2018, pp. 801–806.

[38] B. Golden, L. Bodin, T. Doyle, and W. Stewart Jr, "Approximate traveling salesman algorithms," *Operations Research*, vol. 28, no. 3-part-ii, pp. 694–711, 1980.

[39] E. D. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J. Potvin, "A tabu search heuristic for the vehicle routing problem with soft time windows," *Transportation Science*, vol. 31, no. 2, pp. 170–186, 1997.

[40] M. Gendreau, A. Hertz, and G. Laporte, "New insertion and postoptimization procedures for the traveling salesman problem," *Operations Research*, vol. 40, no. 6, pp. 1086–1094, 1992.

[41] O. Braysy, G. Hasle, and W. Dullaert, "A multi-start local search algorithm for the vehicle routing problem with time windows," *European Journal of Operational Research*, vol. 159, no. 3, pp. 586–605, 2004.

[42] Y. Zhou and J. Wang, "A local search-based multiobjective optimization algorithm for multiobjective vehicle routing problem with time windows," *IEEE Systems Journal*, vol. 9, no. 3, pp. 1100–1113, 2015.

[43] J. Bautista, E. Fernández, and J. Pereira, "Solving an urban waste collection problem using ants heuristics," *Computers & Operations Research*, vol. 35, no. 9, pp. 3020–3033, 2008.

[44] A. M. Benjamin and J. Beasley, "Metaheuristics for the waste collection vehicle routing problem with time windows, driver rest period and multiple disposal facilities," *Computers & Operations Research*, vol. 37, no. 12, pp. 2270–2280, 2010.

[45] K. Buhrkal, A. Larsen, and S. Ropke, "The waste collection vehicle routing problem with time windows in a city logistics context," *Procedia-Social and Behavioral Sciences*, vol. 39, pp. 241–254, 2012.

[46] Y. Shi, L. Lv, F. Hu, and Q. Han, "A heuristic solution method for multi-depot vehicle routing-based waste collection problems," *Applied Sciences*, vol. 10, no. 7, p. 2403, 2020.

[47] B. Crevier, J.-F. Cordeau, and G. Laporte, "The multi-depot vehicle routing problem with inter-depot routes," *European Journal of Operational Research*, vol. 176, no. 2, pp. 756–773, 2007.

[48] E. Babaee Tirkolaee, P. Abbasian, M. Soltani, and S. A. Ghaffarian, "Developing an applied algorithm for multi-trip vehicle routing problem with time windows in urban waste collection: A case study," *Waste Management & Research*, vol. 37, no. 1_suppl, pp. 4–13, 2019.

[49] H. Han and E. Ponce-Cueto, "Waste collection vehicle routing problem: A literature review," *PROMET - Traffic&Transportation*, vol. 27, 08 2015.

[50] M. Rabbani, H. Farrokhi-asl, and H. Rafiei, "A hybrid genetic algorithm for waste collection problem by heterogeneous fleet of vehicles with multiple separated compartments," *Journal of Intelligent & Fuzzy Systems*, vol. 30, no. 3, pp. 1817–1830, 2016.

[51] E. B. Tirkolaee, I. Mahdavi, and M. M. S. Esfahani, "A robust periodic capacitated arc routing problem for urban waste collection considering drivers and crew's working time," *Waste Management*, vol. 76, pp. 138–146, 2018.

[52] H. Wu, F. Tao, and B. Yang, "Optimization of vehicle routing for waste collection and transportation," *International Journal of Environmental Research and Public Health*, vol. 17, no. 14, p. 4963, 2020.

[53] F. Arnold and K. Sorensen, "Knowledge-guided local search for the vehicle routing problem," *Computers & Operations Research*, vol. 105, pp. 32–46, 2019.

[54] F. Arnold, M. Gendreau, and K. Sorensen, "Efficiently solving very large-scale routing problems," *Computers & Operations Research*, vol. 107, pp. 32–42, 2019.

[55] G. Fleury, C. Prins, P. Lacomme, C. Prins, and W. R. Chérif, "Robustness evaluation of solutions for the capacitated arc routing problem," in *Proceedings of the Conference on AI Simulation and Planning in High Autonomy Systems*, 2002, pp. 290–295.

[56] L. Yuan, M. Li, and J. Li, "Research on diversity measure of niche genetic algorithm," in *2008 Second International Conference on Genetic and Evolutionary Computing*. IEEE, 2008, pp. 47–50.

[57] J. Wang, K. Tang, J. A. Lozano, and X. Yao, "Estimation of the distribution algorithm with a stochastic local search for uncertain capacitated arc routing problems," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 1, pp. 96–109, 2015.

[58] K. Maekawa, N. Mori, H. Tamaki, H. Kita, and Y. Nishikawa, "A genetic solution for the traveling salesman problem by means of a thermodynamical selection rule," *Transactions of the Society of Instrument and Control Engineers*, vol. 33, no. 9, pp. 939–946, 1997.

[59] J.-Y. Potvin and S. Bengio, "The vehicle routing problem with time windows part ii: genetic search," *INFORMS journal on Computing*, vol. 8, no. 2, pp. 165–172, 1996.

[60] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Transactions on evolutionary computation*, vol. 4, no. 3, pp. 284–294, 2000.