https://dx.doi.org/10.1016/j.swevo.2020.100730



# An Adaptive Memetic Approach for Heterogeneous Vehicle Routing Problems with Two-Dimensional Loading Constraints

Nasser R. Sabar[1], Ashish Bhaskar[2], Edward Chung[3], Ayad Turky[4], Andy Song[5]

[1]*Department of Computer Science and Information Technology, La Trobe University, Melbourne, Australia*

[2]*Smart Transport Research Centre, Queensland University of Technology, QLD 4001, Brisbane, Australia*

[3]*Department of Electrical Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong*

[4]*College of Engineering and Science, Victoria University, Melbourne, Australia*

[5]*School of Computer Science and Information, RMIT University, Melbourne, Australia*

## Abstract

The heterogeneous fleet vehicle routing problem with two-dimensional loading constraints (2L- HFVRP) is a complex variant of the classical vehicle routing problem. 2L-HFVRP seeks for minimal cost set of routes to serve a set of customers using a fleet of vehicles of different capacities, fixed and variable operating costs, different dimensions, and restricted loading constraints. To effectively deal with the 2L-HFVRP, we propose a two-stage method that successively calls the routing stage and the packing stage. For the routing stage, we propose an adaptive memetic approach that integrates new multi-parent crossover operators with multi-local search algorithms in an adaptive manner. A time-varying fitness function is proposed to avoid prematurity and improve search performance. An adaptive quality-and-diversity selection mechanism is devised to control the application of the memetic op-

*Email addresses:* n.sabar@latrobe.edu.au (Nasser R. Sabar[1]), ashish.bhaskar@qut.edu.au (Ashish Bhaskar[2]), edward.cs.chung@polyu.edu.hk (Edward Chung[3]), ayad.turky@vu.edu.au (Ayad Turky[4]), andy.song@rmit.edu.au (Andy Song[5])

erators and the local search algorithms. In the packing stage, five heuristics are adopted and hybridised to perform the packing process. Experiments on a set of 36 2L-HFVRP benchmark instances demonstrate that the proposed method provides highly competitive results in comparison with state-of-the-art algorithms. In particular, the proposed method obtains the best results for several instances.

*Keywords:* memetic algorithm, adaptive algorithm, vehicle routing, multi-methods

---

## 1. Introduction

The vehicle routing problem (VRP) is one of the fundamental combinatorial optimization problems which is notable for its crucial impact on logistics and freight transportation systems [4], [31]. In a traditional VRP, we are given a set of customers geographically distributed, each with a given demand, and a set of vehicles. The aim is to generate a set of vehicle routes of minimum total cost to serve all customer demands. Taking into account the constraints and requirements of real-world applications, a traditional VRP has been extended into several variants such as the capacitated VRP [31], VRP with time windows [2] and the heterogeneous VRP [13]. In the literature, the traditional VRP and its variants are known as $\mathcal{NP}$-hard problems [31], so meta-heuristic approaches (local search algorithms and population-based algorithms (aka evolutionary approaches)) are very useful to address these problems, as exact methods can only solve small-sized instances [31]. Several survey articles and books have comprehensively reviewed and classified VRP from different perspectives [31] and [18].

In the aforementioned VRP variants, the demand of each customer is defined as a positive integer number which represents the total volume of all items. Thus, the developed solution methodology needs to ensure that the total volume allocated to each vehicle does not exceed its capacity. Hence, the feasibility of a solution can be easily attained. However, in real-world logistic and transportation applications, loading customer items into a vehicle presents a great challenge. Several features and constraints need to be considered when performing the loading process [10]. Consequently, a variant of VRP that takes into consideration the packing process was introduced, known as the VRP with loading constraints [10]. This VRP variant integrates routing with packing during the solving process. It aims to feasi-

bly load customer items into vehicles while attempting to minimise the total cost. Since both routing and packing are well-known $\mathcal{NP}$-hard problems, combining them into the so called VRP with loading constraints leads to an extremely challenging optimisation problem, thus meta-heuristic approaches are highly advisable to solve this problem [18].

A practical and a very challenging variant of the VRP with loading constraints that has received a little attention in the literature is the heterogeneous VRP with two-dimensional loading constraints (denoted as 2L-HFVRP). In 2L-HFVRP, the fleet of vehicles is characterised by different capacities, fixed and variable operating costs, and different dimensions [15]. The demand of each customer is a set of two-dimensional rectangular items characterised by width, length and weight. 2L-HFVRP has a crucial application in transportation and logistics as most of companies are either have a heterogeneous fleet of vehicles or hire different types of vehicles of various capacities to serve their customers.

Similar to the VRP with loading constraints, meta-heuristic approaches are highly appropriate to deal with 2L-HFVRP [15]. However, despite the enormous practical importance of the 2L-HFVRP, a very little research has been done on 2L-HFVRP. Leung et al. [15] introduced the benchmark instances for 2L-HFVRP. The authors applied a local search algorithm for the routing stage and six different heuristics for the packing stage. In [32], the authors proposed a hybrid swarm algorithm that combines artificial bee colony algorithm and artificial immune system for 2L-HFVRP. The proposed hybrid algorithm explores both feasible and infeasible search spaces, and uses simulated annealing as a refinement procedure. The computational results show that the hybrid algorithm is better than the two local search algorithms introduced in [15]. Although the proposed hybrid algorithm is a population-based algorithm, it heavily relies on local search algorithm or local search components. The algorithm did not use evolutionary operators (crossover and mutation) as the main operators to evolve a new population of solutions.

However, despite the good results, it is well-known in the literature that local search algorithms are often work well only when the search space is smooth as they are not robust enough to handle difficult problems [16], [26]. This provided the motivation to develop an evolutionary approach that operates on a population of solutions to effectively solve the larger-sized of 2L-HFVRP instances. To the best of our knowledge, no evolutionary approaches have been developed for 2L-HFVRP, despite of their popularity

in dealing with various hard computational problems [9], [12], [3], [5], [29]. Thus, the aim of this work is to develop an effective adaptive variant of the evolutionary approach, named the memetic approach (MA), for 2L-HFVRP.

MA is a branch of evolutionary approaches (EAs) which has already been proven to be a powerful approach for solving various optimisation problems [16], [33], [20], [1]. It was introduced to alleviate the slow convergence issue which is a common and a very critical issue in most EA variants. The convergence issue is often a twofold one that occurs because EAs use a population of solutions and there is a lack of exploitation (intensification) practice. MA mitigates this issue by leveraging the exploitation capability of the local search (LS) algorithm to compensate for the deficiencies of traditional EAs. Apparently, LS has a huge impact MA performance. Thus, given the diverse characteristics of different optimisation problems, several LSs, featured with distinctive search mechanisms and/or operators, were proposed in the literature. However, due to the dynamic changes of the search landscape, it is very difficult to determine in advance which LS should be used or at which decision point should be applied [16]. Another important design issue that must be taken into consideration when developing an MA or any EA is the configuration (variation or reproduction operators) selection [16]. In the literature different configurations (crossover and mutation operators) have been proposed. Yet, in addition to being problem dependent, the selection of which configuration should be applied is crucial for the MA performance. Improper selection can lead to the so called convergence and diversity issues, which could also be due to the excessive use of LS without taking into consideration the diversity of the search. These issues are common to many other meta-heuristic approaches [16]. This work proposes an adaptive MA that integrate several distinctive features in an adaptive manner to address these issues which are often encountered in pure EAs and MAs.

### *1.1. Goals and Contributions*

The ultimate goal of this work is to develop an effective method for 2L-HFVRP. The proposed method involves the following two stages:

1. **Stage 1: Routing**. We propose an adaptive memetic approach (MA) for the routing problem. The proposed MA is featured by several key components. First, new crossover operators are designed for solutions combination process. The main idea is to guide the search processes by merging high quality and diverse solutions using multi-parent and

multi-crossover operators. Secondly, a pool of local search algorithms is adopted to improve the generated solutions. The key idea is to combine the complementary features of various local search algorithms in an adaptive manner to effectively navigate the search space around the generated solutions. Thirdly, to avoid prematurity and improve the search performance, we propose a time varying fitness function that allows the search to focus on exploration in the early stages and then gradually decreases so that the search focuses more on exploitation. Finally, an adaptive quality-and-diversity selection mechanism is devised to control the application of memetic operators and the local search algorithms are able to find the proper balance between the exploration and exploitation ability of each operator.

2. **Stage 2: Packing**. In this work, we adopted five different heuristics to perform the packing process. Since different vehicle routes represent a sub-packing problem, it is not known in advance which heuristic should be used first. Calling them one by one would be computationally expensive and might not lead to a feasible solution. Thus, in this work, the packing heuristics are randomly hybridised with each other. Our idea is to combine the complementary strengths of different packing heuristics in order to effectively handle various packing scenarios.

The main contributions of this paper can be summarised as follows:

- An adaptive memetic approach that uses various evolutionary operators and multi-local search algorithms is proposed to solve 2L-HFVRP more effectively.
- New multi-parent crossover operators for solutions combination process that merge high quality and diverse solutions.
- A time-varying fitness function to void prematurity and improve the search performance.
- An adaptive quality-and-diversity selection mechanism to control the application of MA operators and the multi-local search algorithms.
- A set of algorithms have been tested on 2L-HFVRP benchmark instances and compared with state-of-the-art algorithms from the literature. Good performance has been demonstrated.

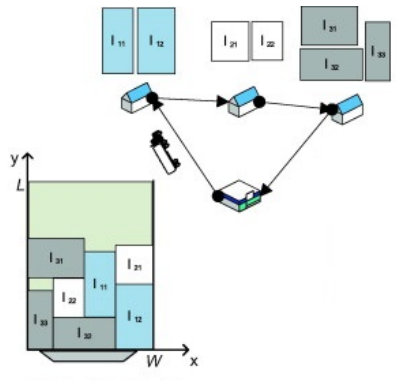


Figure 1: An example of 2L-HFVRP adopted from [15]

*1.2. Evaluation*

We evaluate the performance of the proposed MA using 36 2L-HFVRP benchmark instances. The computational results are compared with different counterparts and the state-of-the-art algorithms.

*1.3. Organisation*

The rest of the paper is organised as follows. Section 2 presents the formulation of the 2L-HFVRP. In Section 3, we describe the proposed method, along with main components. Section 4 discusses the experimental step. The computational results and comparison are provided in Section 5. Finally, we conclude the paper in Section 6.

## 2. Problem description

This work focuses on the heterogeneous fleet vehicle routing problems with two-dimensional loading constraints (2L-HFVRP). In 2L-HFVRP, we have heterogeneous fleet of vehicles of different types that are initially located at the depot (central point) and a group of customers with known demands. The demand of each customer is a set of two-dimensional rectangular items characterised by width, length and weight [15]. The objective is to design a least cost set of vehicle routes to serve all customers. An example of 2L-HFVRP with three customers of different items and one vehicle route is shown in Figure

The generated solution (the set of vehicle routes) is feasible if the following routing and packing constraints are satisfied [15]:

- Items of the same customer must be packed into the same vehicle.
- All items must be loaded with edges parallel to the edge of the vehicle.
- Overlap between items within the same vehicle is not allowed.
- Each customer must be served only once by exactly one vehicle.
- All vehicles must start and terminate their routes at the depot.
- The total demand assigned for each vehicle should not exceed the vehicle capacity, length and width.

Formally, 2L-HFVRP can be modelled as an undirected connected graph, $G(V, E)$, where $V = \{v_0, v_1, \ldots, v_n\}$ is a set of nodes representing the depot ($v_0$) and the group of customers ($v_1, \ldots, v_n$) [15]. $E = \{(v_i, v_j) : v_i, v_j \in V, i < j\}$ is an edge set that connects customers with each other and with the depot. Each edge $E$ is associated with a non-negative value which denotes the cost (the travel time or distance) defined by a matrix $C = (c_{ij})$, where $c_{ij}$ represents the cost between customers $v_i$ and $v_j$. $P$ is a fleet of heterogeneous vehicles of different types. Each type $t$ ($t = 1, \ldots, P$) of the fleet of vehicles has an unlimited number of vehicles and differentiated by capacity $Q_t$, variable cost $V_t$, fixed cost $F_t$, length $L_t$ and width $W_t$. The loading area of each vehicle of each type $t$ is $A_t$=$W_t \times L_t$. Given the fact that a vehicle with a higher weight-loading cost has a higher fixed and variable cost, it is assumed that $Q_1 \leq Q_2 \leq \cdots \leq Q_P$, $F_1 \leq F_2 \leq \cdots \leq F_P$, and $V_1 \leq V_2 \leq \cdots \leq V_P$. Each customer $i$ ($i = 1, \ldots, n$) is associated with a non-negative value which represents the requested set of $m_i$ rectangular items denoted as $IT_i$ to be delivered. The total weight of $IT_i$ is denoted as $D_i$. The length and width of each item $I_{ir} \in IT_i (r = 1, 2, \ldots, m_i)$ are denoted as $l_{ir}$ and $w_{ir}$. The total area of all items requested by customer $i$ is $a_i$=$\sum_{r=1}^{m_i} w_{ir} \times l_{ir}$. The cost of each route ($R$) of vehicle type $t$ is calculated is follows [15]:

$$\Pi_R = F_t + \sum_{i=1}^{i \leq |R|} V_t \times c_{R(i),R(i+1)} \tag{1}$$

and the cost for a solution (objective function or fitness function ($f$)) is calculated using Equation (2).

$$f = \sum_{i=1}^{K} \Pi_{R_i} \tag{2}$$

where $K$ is the total number of routes in a given solution.

## 3. Methodology

Our proposed method for 2L-HFVRP consists of two stages, routing and packing, as shown in Figure 2. It takes the given 2L-HFVRP instance as an input and then calls the routing stage (Section 3.1) to generate a 2L-HFVRP solution that comprises a set of vehicle routes, followed by the packing stage (Section 3.2) to load customer items into the assigned vehicle. The routing and packing stages are successively executed until satisfying the pre-defined termination condition.

We further discuss these two stages along with the proposed key components in the following subsections.

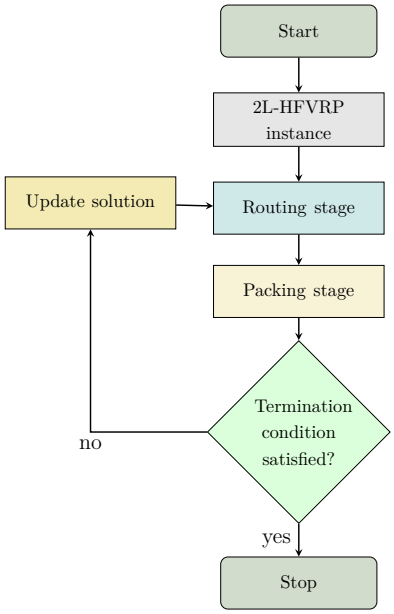


Figure 2: The flowchart of the proposed method

### *3.1. Routing stage*

The routing stage is responsible for generating a solution for the 2L-HFVRP instance. The solution involves a set of vehicle routes in which each route serves one or a group of customers. Success in generating a good solution strongly depends on the solution generation methodology which in turn relies on its internal components. In this work, we propose an adaptive memetic approach (MA) to find high quality solutions to 2L-HFVRP. The proposed MA integrates several distinctive features in an adaptive manner to address these issues which are often encountered in pure EAs and MAs. Firstly, we propose new multi-parent crossover operators for solutions combination process to merge high quality and diverse solutions. Secondly, we utilise a pool of LS algorithms to effectively explore the areas around the generated solution. Thirdly, we propose a time-varying fitness function to void prematurity and improve the search performance. Finally, an adaptive quality-and-diversity selection mechanism is devised to control the application of the MA configuration and the LS algorithms.

Figure 3 shows the working procedure of the proposed MA. It follows the general paradigm of a traditional MA which hybridises an EA variant with a LS. It uses genetic algorithm (GA) where a LS is performed before it moves on to the next generation, so that the search space around the current solution is effectively explored. It first sets the parameters and then creates a population of solutions. It then assigns a fitness value for each solution in the population by using the proposed time-varying fitness function described in Section 3.1.4. Then, it executes the main evolutionary process to improve the population of solutions for a fixed number of generations. Each generation evolves a new set of solutions to replace the old ones using the evolutionary operators. MA first calls the selection mechanism (Section3.1.5) to form the mating pool and then executes the proposed multi-parent crossover operators (Section 3.1.6) to generate a new offspring. The offspring is then mutated (Section 3.1.7) and further improved by the LS algorithm (Section 3.1.8). Finally, it checks the stopping condition (Section 3.1.9) and calls the population update step to decide if the improved offspring can be added into the population. In following subsections, we introduce the key elements of our MA.

#### *3.1.1. Solution representation*

In our MA, each solution (chromosome or individual) is a permutation of a set of customers. The permutation represents the visiting order of each customer in a given solution and it is represented as a one-dimensional array

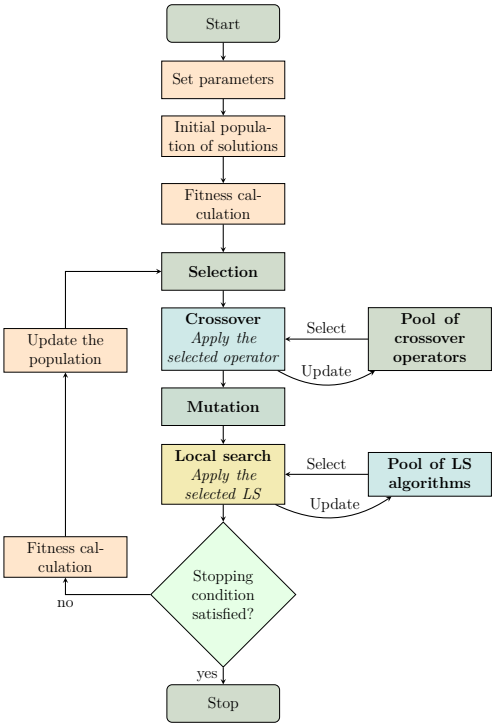


Figure 3: The flowchart of the proposed MA

without route delimiters, as shown in Figure 4. The size of the array is equal to the maximum number of customers in a given 2L-HFVRP instance. We use this representation as it allows various crossover operators to be simply applied without considering the infeasibility issue [19]. To convert the given solution into a set of vehicle routes, we use a split procedure to cut the solution into a set of routes based on vehicle capacity [19].

| $C_2$ | $C_1$ | $C_4$ | $C_3$ | $C_6$ | $C_8$ | $C_5$ | $C_9$ | $C_7$ |
|---|---|---|---|---|---|---|---|---|

Figure 4: Solution representation

*3.1.2. Set the parameters*

The proposed MA has four parameters that need to be set by the user. These are:

- Maximum number of generation ($MaxGen$)—represents the maximum number of generations to be performed by MA.

- Population size ($PS$) —represents the number of solutions to be saved in the population.
- Crossover rare ($CR$) —indicates the probability of applying crossover operator.
- Mutation rate ($MR$) —indicates the probability of calling mutation operator.

We conducted a preliminary experiment to set these parameters (see Section 4)

*3.1.3. Create a population of solutions*

The proposed MA uses a randomised method to initialise the population of solutions. Starting from an empty solution and from the first cell in the solution, the method iteratively and randomly picks an unallocated customer and assigns it to the current cell. This process is repeated to initialise $PS$ solutions. Algorithm 1 shows the pseudocode of the population initialisation method.

**Algorithm 1:** Population initialisation method

```
PS /*Population size*/;
POP /*Population of solutions*/;
N_c= Total number of customers in a given instance;
Set_c= {Set of customers in a given instance};
for i=1 to PS do
    S= empty solution;
    for j=1 to N_c do
        C= Random ∈ Set_c;
        S_{i,j}=C;
        Remove C from Set_c;
    end
    Add S to POP;
end
Output final population, POP;
```

*3.1.4. Fitness calculation*

Fitness calculation (aka the cost or the quality) uses the so called fitness function (or the objective function) to assign a value to each solution in the population. A solution in a given population is characterized by its fitness value which demonstrates how good this solution is compared to the other ones. Obviously, the fitness function is largely responsible for guiding the

optimisation algorithm. In many EAs and other search methods the fitness functions rely almost exclusively on the fitness value (cost) of the solution. This kind of fitness function does not take into consideration the diversity of the search, which has a crucial impact on the search performance. For instance, in 2L-HFVRP and many other problems (combinatorial or continuous), there could be a situation in which there are two or more solutions with a different sequence (or value) of decision variables but they return the same fitness value, i.e., two solutions might have different genotypes but the same phenotypes. This indicates that the information provided by the fitness function may not be sufficient to guide the search process as several solutions from different areas could be discarded because their fitness values are same as the current ones.

To alleviate this issue, this work proposes a time varying fitness function that evaluates both the fitness and diversity of each solution in the population. It allows the proposed MA to focus on exploration in the early stages and then gradually decreases so that the search will focus more on exploitation. This will not only help the proposed MA to effectively explore new areas in the solution search space but it can also help to escape from the basin of attraction points, thus avoiding stalling at the local optima. Given a solution $S$ ($S \in Pop$), the proposed fitness function ($f$) calculates its fitness value as follows:

$$minf(S) = \Omega_1 \times \Phi(S) + \left(\frac{1}{\Psi(S)}\right) \times \Omega_2 \tag{3}$$

Equation (3) has two parts: the cost value ($\Phi$) of $S$ which is calculated using Equation (2) and the diversity value ($\Psi$) of $S$. $\Omega_1$ is a time varying variable that will be linearly varying based on the current generation as follows:

$$\Omega_1 = \begin{cases} vt & \text{if } vt > 0 \\ 1 & \text{otherwise} \end{cases}$$

and

$$vt = (max_{vt} - min_{vt}) \times \frac{MaxGen - Gen}{MaxGen} + min_{vt} \tag{4}$$

where $min_{vt}$ and $max_{vt}$ are constants that represent the minimum and the maximum time variation. $Gen$ is the current generation of MA and $MaxGen$

is the maximum number of generations. $\Omega_1$ starts with a large value, and then decreases as MA progresses. $\Omega_2$ forces MA to focus on improving the cost ($\Phi$) only at the late stages of the search process as follows:

$$\Omega_2 = \begin{cases} 0 & \text{if } \Omega_1 = 1 \\ 1 & \text{otherwise} \end{cases}$$

The diversity value of $S$ ($\Psi(S)$) represents the average diversity between $S$ and all members in the current population as follows:

$$max\ \Psi(S) = \frac{1}{PS} \sum_{i=1}^{PS} dis(S, c_i), \forall c \in POP \tag{5}$$

where $dis(S, c)$ is the dissimilarity between $S$ and the solutions in the population in terms of how many customers have not been assigned to the same position in the same route in both solutions. It is calculated as follows [14]:

$$dist(s_1, s_2) = \frac{|E(s_1) \bigcup E(s_2)| - |E(s_1) \bigcap E(s_2)|}{|E(s_1) \bigcup E(s_2)|} \tag{6}$$

where $E$ represents the number of edges. Two solutions are considered to be similar if the distance between them is zero.

*3.1.5. Selection*

The proposed MA uses a selection mechanism to choose solutions from the current population to be added into the mating pool. We use the binary tournament selection mechanism to form the mating pool. It works as follows: select two solutions randomly from the population and insert the best solution into the mating pool. Repeat this process $N$ (e.g., $N = \frac{PS}{2}$) times.

*3.1.6. Crossover*

In EAs or MAs, the crossover operator is the most salient operator whose main goal is to guide the search process towards a new promising regions in the search space. It takes more than one parent solutions and then mixes their genetic material to produce an offspring solution, looking for a better solution. According to the literature, a crossover operator may be particularly suited to a given problem instance or may only work well during a certain states of the search process [11] [17]. Consequently, a number of different

crossover operators have been proposed. Of these, multi-parent operators have been shown to produce better results than traditional crossover operators for various optimisation problems [7]. It combines the genetic material of more than two parents to generate a new offspring. However, in spite of this success, a very few papers are available on multi-parent operators for combinatorial optimisation problems, especially, routing and scheduling problems [17]. In addition, most of existing multi-parent operators do not use any kind of knowledge to guide the selection process where all parent solutions are selected via the traditional mechanism [17].

In this work, we propose four different multi-parent crossover operators for MA. The proposed operators exploit knowledge obtained in previous generations to guide the search to produce a high quality and diverse offspring for the next generation. Each operator uses three parent solutions of different characteristics. The first and second parent represent the diversity element and selected via the selection mechanism described in Section 3.1.5. The third parent solution constitutes the quality element and is generated by extracting knowledge from all solutions of the current population. The rationale behind this is that the extracted knowledge can be very useful for guiding MA towards promising areas in the search landscape. One of the issues encountered when using MA is the inability to know a priori which operator performs best for a given problem. Selecting a crossover operator that yield good performance has been an active research area in the literature [11]. We therefore use an on-line selection mechanism to determine which operator should be used at each decision point. By using multi-crossover operators, the proposed MA can effectively deal with various problem instances and cope with the search landscape changes. Our ultimate aim is to combine the complementary strengths of various crossover operators within MA. The working steps of the proposed multi-parent crossover operators are:

- **Step 1:** Call the selection mechanism (Section 3.1.5) to select the first and second parent solutions, $P_1, P_2$.

- **Step 2:** Use the frequency matrix to generate the third parent solution, $P_3$.

- **Step 3:** Call the selection mechanism to select a crossover operator from the pool of operators.

- **Step 4:** Apply the selected crossover operator on $P_1$, $P_2$ and $P_3$.

- **Step 5:** Update the frequency matrix and the impact of the selected operator.

Each of these components of the proposed crossover are described in more detail below.

- **Frequency matrix**. The frequency matrix saves the frequency of assigning a customer to the same location across all solutions in the current population. That is, it stores how many times a customer has been assigned to the same location or position. The stored values are extracted from the current population of solutions, which are updated every generation. Initially, all values of the matrix are zero. Figure 5 displays an example of a frequency matrix. In the figure, columns represent the location index, while rows represent the customer index. In this example, the matrix involves five customers and five locations. The matrix can be read as follows: customer 1 has been assigned to location 1 four times, to location 2 two times, to location 3 six times, to location 4 three times and to location 5 once; and so on for the other customers. We then transform the extracted frequencies (extracted knowledge) of each customer into a selection probability and use the roulette wheel ($RW$) selection mechanism to generate a new solution. In $RW$, the probability ($P$) of $i^{\text{th}}$ customer to be copied into the $j^{\text{th}}$ position of the new solution is proportional to its total frequency value ($f$) at all $j^{\text{th}}$ positions in current population, which is calculated as follows:

$$P(i) = \frac{f(i)}{\sum_{j=1}^{PS} f(j)}, i \neq j \tag{7}$$

  where $PS$ is the population size.

- **Pool of crossover operators**. The proposed multi-parent crossover operators ($CXs$) use three parents ($P_1$, $P_2$ and $P_3$) to generate an offspring solution. Our proposed operators mix various complementary characteristics of different operators in effective ways to preserve quality and diversity. $P_1$ and $P_2$ are selected from the current population by the selection mechanism (see Section 3.1.5), while $P_3$ is generated via the frequency matrix (knowledge extraction) process. We propose four different operators (denoted as $CX_1$ to $CX_4$):

| | location index | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| 1 | 4 | 2 | 6 | 3 | 1 |
| 2 | 1 | 1 | 3 | 5 | 5 |
| 3 | 3 | 4 | 2 | 1 | 2 |
| 4 | 4 | 5 | 4 | 2 | 4 |
| 5 | 3 | 6 | 1 | 4 | 3 |

Figure 5: Frequency matrix

1. $CX_1$: Randomly select a segment of consecutive positions from Parent 3 ($P_3$) and move them following their index into the offspring. The remaining positions are randomly selected from either $P_1$ or $P_2$; the probability of selection is 0.5 for both.
2. $CX_2$: Randomly select a segment of consecutive positions from the middle of $P_3$ and move them into the offspring. The remaining positions on the left side are selected from $P_1$ and the right side are selected from $P_2$.
3. $CX_3$: The positions of an offspring are selected from $P_3$, $P_1$ and $P_2$ with the probabilities $P_{p1}$=0.2, $P_{p2}$=0.2 and $P_{p3}$=0.6, respectively.
4. $CX_4$: The positions of an offspring are selected from $P_3$ and their adjacent from $P_1$ and $P_2$ as follows: select the first position from $P_3$ and then fills in the second position using the adjacent of the first position in $P_1$ and the third position using the adjacent of the first position in $P_2$. The process is repeated for all other positions in the offspring.

It should be noted that the above crossover operators always generate a feasible offspring as they discard duplicated positions and select only missed ones.

- **Operator selection**. The operator selection strategy chooses one crossover operator from the given set based on their previous impact values. In this work, the impact value of each operator represents the summation of the diversity and quality, described in the following subsection. We use the multi-armed bandit mechanism (MAB) as the selection strategy [8]. Formally, let $p_i$ represents the impact value of

$i^{\text{th}}$ operator, $n_i$ is the number of times that the $i^{\text{th}}$ operator has been used so far, MAB selects the operator that maximises Equation (8) [8].

$$max_{i=1...N_{op}} \left( p_{i(t)} + c\sqrt{\frac{2\log \sum_{j=1}^{N_{op}} n_{j(t)}}{n_{i(t)}}} \right), \tag{8}$$

where $t$ is the current generation, $N_{op}$ is the number of operators and $c$ is the scaling factor.

- **Operator impact update**. The operator impact maintains a value for each operator that represents the performance of this operator. The values of all operators are used by the operator selection strategy to decide which one should be used. In this work, we use a reward-punishment scheme to assess the impact of the applied operator. An operator is rewarded if it improves the solution; otherwise it is punished. Let $r_{(t)}$ be the impact value of $i^{\text{th}}$ operator at iteration $t$. The reward-punishment scheme works as follows:

$$r_{i(t)} = r_{i(t)} + (\triangle + D) \tag{9}$$

and

$$r_{j(t)} = r_{j(t)} - (\triangle/(N_{op} - 1)), \forall j \in \{1, 2, ..., N_{op}\} \text{ and } i \neq j, \tag{10}$$

where

$$\triangle = \frac{f_1 - f_2}{f_1 + f_2} \tag{11}$$

Otherwise (if $i^{\text{th}}$ operator cannot improves the solution):

$$r_{i(t)} = r_{i(t)} - |(\triangle * It)| \tag{12}$$

and

$$r_{j(t)} = r_{j(t)} + (|\triangle| * It/(N_{op} - 1)), \forall j \in \{1, 2, ..., N_{op}\} \text{ and } i \neq j, \tag{13}$$

where $It = current\_generation/\ total\_generations$ and $N_{op}$ represents the number of operators. $f_1$ is the fitness of the best among the selected parent calculated using Equation (2). $f_2$ is the fitness of the solution generated by the applied operator calculated using Equation (2). $D$ is the diversity value which represents the average differences between the generated solution and the selected parent calculated using Equation (5). The probabilities of all operators initialised to $1/N_{op}$.

*3.1.7. Mutation*

The mutation operator induces a diversity into the search in order to help MA to escape from the basin of attraction points. A solution will be mutated if the generated random number is less than the mutation rate ($MR$). We use the swap mutation operator to exchange the positions of two randomly selected customers.

*3.1.8. Local search*

The local search (LS) algorithm is a key element in MA which focuses on accelerating the convergence of the search process. Generally, two important considerations should be taken into account when using LS within MA [16], [23], [27], [21]. First, applying LS at every generation could be computationally expensive and might increase the risk of being too exploitive [22], [24]. Second, given the diverse characteristics of different problem instances, no single LS performs consistently across all instances [25], [28]. Therefore, to mitigate these issues, we control the LS application frequency and utilise a pool of LS algorithms. That is, we only apply an LS if the current solution cannot be added into the population and if so, we use a selection strategy to choose one LS from the given pool to improve the current solution. In what follows, we discuss the pool of LSs and LS selection strategy.

- **Pool of LSs**. Our pool of LSs contains three LSs (denoted as $LS_1$, $LS_2$ and $LS_3$). Each one uses different acceptance rules to navigate the search space. All LSs start with an initial solution and then successively call the following two components: a move operator to generate a new solution and the acceptance rule to decide whether to accept or reject the generated solution. The utilised move operator randomly shifts one customer from its current location to another one. The acceptance rules of the utilised LSs are [30]:

1. $LS_1$: Steepest descent. The new solution $S_1$ will be accepted if it has a better quality than the initial solution $S_0$. $LS_1$ will be halted after 10 non-improving iterations (fixed based on a preliminary test).
2. $LS_2$: Great deluge. The generated solution $S_1$ will be accepted if it has a better quality than the initial solution $S_0$ or lower than the acceptance level ($level$). The initial value of the $level$ is equal to the quality of the initial solution $S_0$. After each iteration of $LS_2$, the $level$ will be decreased as follows: $level$=$level$-$\varepsilon$, where $\varepsilon$= ($f(S_1)$-$f(best_{sol})$)/$It$. $It$ is the maximum number of iterations which is fixed into 1000 (determined based on a preliminary test). $LS_2$ will be terminated if the $level$ value is lower than the best solution found so far.
3. $LS_3$: Simulated annealing. The generated solution $S_1$ will only be accepted if it has better quality than the initial solution $S_0$ or it satisfies the acceptance probability ($P$), $R$ <$P$ where $R$ is random number between [0,1] and $P$ is calculated as follows: $P$= $\exp(-\delta/t)$, $\delta$ is the difference between quality of $S_1$ and $S_0$, $t$ is the temperature initially set into 50% of the quality value of $S_0$ and decreased after each $LS_3$ iteration by $\alpha$ ($\alpha$=0.85). $LS_3$ will stop when $t$=0.

- **LS selection strategy**. The selection strategy uses the historical improvement of all LSs to decide which one should be used. It successively invokes the following process:

  – LS selection. In this work, we use the MAB as our LS selection mechanism, see Equation (8). MAB uses the information provided by the LS impact evaluation to select one LS from the pool.

  – LS Impact evaluation. The impact evaluation saves and updates the information related to each LS performance. It has two counters: $LS_{app}$ is the number of times each LS has been used and $LS_{per}$ is the accumulated percentage improvement of each LS. $LS_{per}$= $(\frac{S_1-S_0}{S_0})$ *100.

*3.1.9. Stopping condition*

The stopping condition is responsible for terminating the search process of MA. It checks whether the maximum number of generations has been

reached. If yes, it terminates MA and returns the best found solution. Otherwise, it calls the following:

1. Fitness calculation: calculate the fitness values of new solutions.
2. Update the population: replace new solutions with old ones if they have better fitness values.

*3.2. Packing stage*

In this work, the packing stage is executed for each solution generated by the routing stage to load all items into the vehicle. It should be noted that the routing stage ensures that the total weight of items requested by customers of each route does not exceed vehicle capacity. Our packing stage involves two processes: arrangement and selection. The arrangement process uses one or two criteria to sort all items. In this work, items are sorted either randomly or by the decreasing surface area. The selection process takes the sorted items and then loads them one by one into the vehicle. In the literature, various heuristics were used for the loading process. Five well-known loading heuristics are:

- $H_1$:Bottom-left fill heuristic —selects the position with minimum $W$-axis coordinates.
- $H_2$: Bottom-left fill heuristic —selects the position with minimum $L$-axis coordinates.
- $H_3$: Maximum touching perimeter heuristic —selects the positions that have maximum value of the summation for all common edges between the current item, the loaded items in the vehicle, and the loading surface of the vehicle.
- $H_4$: Maximum touching perimeter no walls heuristic —selects the positions that have maximum value of the summation for all common edges between the current item and the loaded items in the vehicle.
- $H_5$: Min area heuristic —selects the positions that have the minimum rectangular surface.

In the literature, these heuristics are applied in a sequential manner. However, different vehicle route often represents a different sub-packing problem which might need a different heuristic or a combination of heuristics. To this end, the five heuristics are randomly hybridised with each other. Our idea is to combine the complementary strengths of different packing heuristics to effectively handle various packing scenarios. We first randomly pick up two or three heuristics and then, for each heuristic, randomly assign a loading proportion. For instance, if we chose two heuristics ($H_1$ and $H_2$), then the proportion of $H_1$ is randomly assigned a value between 5% to 90%, and the reset will be for $H_2$. If $H_1$ is allocated to 60% and $H_2$ to 40%, then this indicates that the loading process will use $H_1$ to load 60% of the items and $H_2$ for the reset (40% of the items). The selection of heuristics and the proportion assignments are invoked every generation.

## 4. Experimental setup

This section consists of two subsections. The first subsection discusses the 2L-HFVRP benchmark instances, while the second one reports the parameter settings of the proposed MA.

### *4.1. Benchmark instances*

We use the 2L-HFVRP benchmark instances introduced in [15] to assess the performance of our MA. The benchmark involves 36 instances of diverse characteristics and each one has five classes; making 180 instances in total. The main characteristics of all instances are tabulated in Table 1. In the table, $m_i$ is a uniformly distributed within the specified ranges. $Vertical$, $Homogeneous$ and $Horizontal$ represent the item shapes. Each item is assigned to one shape with equal probability and the corresponding item sizes are randomly generated within the given ranges. The number of customers ranges from 15 to 255 whereas the number of items ranges from 15 to 786. More details on 2L-HFVRP benchmark instances can be found in [15].

Table 1: The characteristics of 2L-HFVRP benchmark instances

| Class | $m_i$ | Vertical | | Homogeneous | | Horizontal | |
|---|---|---|---|---|---|---|---|
| | | Length | Width | Length | Width | Length | Width |
| 1 | [1,2] | [0.4L,0.9L] | [0.1W,0.2W] | [0.2L,0.5L] | [0.2W,0.5W] | [0.1L,0.2L] | [0.4W,0.9W] |
| 2 | [1,3] | [0.3L,0.8L] | [0.1W,0.2W] | [0.2L,0.4L] | [0.2W,0.4W] | [0.1L,0.2L] | [0.3W,0.8W] |
| 3 | [1,4] | [0.2L,0.7L] | [0.1W,0.2W] | [0.1L,0.4L] | [0.1W,0.4W] | [0.1L,0.2L] | [0.2W,0.7W] |
| 4 | [1,5] | [0.1L,0.6L] | [0.1W,0.2W] | [0.1L,0.3L] | [0.1W,0.3W] | [0.1L,0.2L] | [0.1W,0.6W] |

*4.2. Parameter settings*

The proposed MA has a few tunable parameters that need to be set by the user. The parameter settings of MA were fine-tuned by conducting experiments on a set of instances for different sets of parameter values. To this end, we did a preliminary investigation to set these parameters. Ten different instances were used to test various parameter values by running the proposed MA 31 times using different combinations of parameter values. Taking into consideration the trade-off between computational time and solution quality, the parameter values that lead to the best cost values were chosen. The tuning process has revealed that population size is the most sensitive parameter concerning computational time. Using a big population size slightly improves the best cost value but dramatically increases the computational time. The configurations of search operators and parameters have shown to be not very sensitive. This is because the proposed approach adaptively changes the configurations and the parameters settings if the current one did not perform well in the past generation. The suggested values, along with the tested ranges, are reported in Table 2.

Table 2: The parameter settings of MA

| **Parameter** | **Tested range** | **Suggested value** |
|---|---|---|
| Maximum number of generations | 5-300 | 150 |
| Population size, $PS$ | 5-40 | 20 |
| Crossover rate, $CR$ | 0.1-0.9 | 0.8 |
| Mutation rate, $MR$ | 0.1-0.9 | 0.03 |
| Maximum number of non-improving iterations for local search | 5-50 | 10 |
| The scaling factor, $c$ | 2-20 | 8 |

## 5. Results and comparison

This section reports the computational results obtained by the proposed MA. Our aims are to:

1. Examine the benefits of the proposed key elements on the search performance.

2. Compare the results of the proposed MA with the state-of-the-art algorithms.

*5.1. Effectiveness evaluation*

This section examines the benefit of the proposed elements on the search performance. Thus, to assess the impact of each of the integrated element, ten different algorithms are implemented and tested. We first test the traditional genetic algorithm (GA) that uses the one-point crossover operator, the swaps-based mutation operator and the tournament selection mechanism. We then add the proposed elements one by one into GA to evaluate their impact on the search performance. The resultant ten different variants are:

- MA: the new algorithm being proposed;
- GA: the foundation of MA — the traditional genetic algorithm;
- $GA_{FT}$: a traditional GA that uses the proposed fitness function only;
- $GA_{CX_1}$: a traditional GA that uses the first crossover operator only;
- $GA_{CX_2}$: a traditional GA that uses the second crossover operator only;
- $GA_{CX_3}$: a traditional GA that uses the third crossover operator only;
- $GA_{CX_4}$: a traditional GA that uses the fourth crossover operator only;
- $GA_{LS_1}$: a traditional GA that uses the first local search only;
- $GA_{LS_2}$: a traditional GA that uses the second local search only; and
- $GA_{LS_3}$: a traditional GA that uses the third local search only.

To ensure a fair comparison, all algorithms are tested using the same computational resources, seed number, parameter values and initial solutions. All algorithms (MA, GA, $GA_{FT}$, $GA_{CX_1}$, $GA_{CX_2}$, $GA_{CX_3}$, $GA_{CX_4}$, $GA_{LS_1}$, $GA_{LS_2}$ and $GA_{LS_3}$) are executed 31 independent runs. As the total number of instances for the 2L-HFVRP is large, the results are presented in an aggregated form as in the literature. The results obtained from all algorithms are presented and compared in Table 3. In the table, we indicate in boldfont the best obtained results. As can be seen from the table, MA obtained the best results across all tested instances. This clearly justifies the benefit of the proposed elements on the search performance of a traditional GA. To further verify the effectiveness of MA, we have conducted a pairwise

comparison between MA and other algorithms using the Wilcoxon statistical test with a significance level of 0.05. The $p$-values are shown in Table 4 where the sign + indicates MA is statistically better than the compared variant. As can be seen from the table, MA is statistically better than the other algorithms (MA $p$-values is less than 0.05). This positive result again shows that MA is an effective methodology for 2L-HFVRP.

We now turn our attention to an evaluation of the each of the proposed elements. We first calculate the improvement percentage (%$\triangle$) between the results of each of the tested algorithm (MA, $GA_{FT}$, $GA_{CX_1}$, $GA_{CX_2}$, $GA_{CX_3}$, $GA_{CX_4}$, $GA_{LS_1}$, $GA_{LS_2}$ and $GA_{LS_3}$) and the GA results as follows: %$\triangle$=(($A1$-$A2$)/$A1$)*100, where $A1$ is the best results obtained by GA and $A2$ is the best results returned by the compared algorithm. The %$\triangle$ of all tested algorithms over GA results are plotted in Figure 6, Figure 7 and Figure 8. A point on the figure indicates the percentage gap between the best result obtained by GA and the best value achieved by the compared algorithm for each instance. The higher the point, the bigger the percentage gap is. The bigger percentage gap indicates that the result of the compared algorithm is better than GA. Observing all figures, we can see that all variants are better than the traditional GA.

We now analyse the results of each sub-figure. From Figure 6, we can see that $GA_{FT}$ improves the GA results on all tested instances. In Figure 7 we compares four GA variants ($GA_{CX_1}$, $GA_{CX_2}$, $GA_{CX_3}$, $GA_{CX_4}$) that use the proposed crossover operators. It can be seen that the results of $GA_{CX_1}$, $GA_{CX_2}$, $GA_{CX_3}$ and $GA_{CX_4}$ are better than GA on all instances. A comparison of the GA variants that employ local search algorithms $GA_{LS_1}$, $GA_{LS_2}$ and $GA_{LS_3}$ is shown in Figure (8). The figure demonstrates that all local search algorithms have an impact on the search performance of GA as all of them improved the results across all instances compared to GA. From the figures (Figures 6, 7 and 8), we can see that no one variant can be considered to be the best across all instances. This is also verified by the statistical comparison. Indeed, each one excels on certain instances or at certain stages of the solving process. This result is also consistence with the No Free Lunch theorem, as can be seen in Figures 9 and 10. The findings from the results analysis of the tested GA variants clearly shows the benefit of combining several elements in an adaptive manner with MA so they can complement each other and excel on all the tested instances.

Table 3: The computational results of the compared variants

| **Inst** | $MA$ | $GA$ | $GA_{FT}$ | $GA_{CX_1}$ | $GA_{CX_2}$ | $GA_{CX_3}$ | $GA_{CX_4}$ | $GA_{LS_1}$ | $GA_{LS_2}$ | $GA_{LS_3}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **587.2** | 724.3 | 683.4 | 612.2 | 597.4 | 594.1 | 596.7 | 590.1 | 589.7 | 591.4 |
| 2 | **676.4** | 811.2 | 747.3 | 714.4 | 722.7 | 731.6 | 718.4 | 686.5 | 693.3 | 692.7 |
| 3 | **758.5** | 904.5 | 844.7 | 800.3 | 811.5 | 821.5 | 842.7 | 780.2 | 786.5 | 794.3 |
| 4 | **677.7** | 801.7 | 790.2 | 768.1 | 771.3 | 744.3 | 756.2 | 681.4 | 694.3 | 697.5 |
| 5 | **783.4** | 993.2 | 811.4 | 806.5 | 801.2 | 820.1 | 811 | 788.3 | 795.7 | 788.8 |
| 6 | **826.2** | 1001.6 | 927.3 | 909.8 | 894.2 | 881.6 | 864.2 | 837.8 | 844.6 | 851.5 |
| 7 | **5323.04** | 6910 | 6608 | 6121 | 6110 | 6181 | 6111 | 6017 | 5891 | 5901 |
| 8 | **5364.01** | 7694 | 7510 | 7343 | 7356 | 7311 | 7313 | 5780 | 5644 | 5687 |
| 9 | **1032** | 1438 | 1386 | 1273 | 1287 | 1271 | 1275 | 1120 | 1122 | 1160 |
| 10 | **6891.14** | 8816 | 8811 | 8635 | 8621 | 8545 | 8627 | 7857 | 7868 | 7892 |
| 11 | **7784** | 9615 | 9107 | 9104 | 9119 | 9083 | 9115 | 8744 | 8625 | 8711 |
| 12 | **1666.86** | 1982 | 1816 | 1797 | 1811 | 1774 | 1782 | 1701 | 1691 | 1698 |
| 13 | **25257.08** | 31467 | 30291 | 29863 | 29712 | 29675 | 29754 | 28437 | 27445 | 26341 |
| 14 | **10668** | 12011 | 11557 | 11274 | 11006 | 11173 | 11284 | 10986 | 10984 | 10957 |
| 15 | **11083.63** | 12624 | 12342 | 12165 | 12282 | 12026 | 12044 | 11875 | 11732 | 11567 |
| 16 | **1280.22** | 1642 | 1575 | 1482 | 1475 | 1498 | 1477 | 1387 | 1364 | 1366 |
| 17 | **1754.84** | 2072 | 1980 | 1926 | 1875 | 1812 | 1797 | 1790 | 1780 | 1785 |
| 18 | **5400** | 7318 | 7295 | 7011 | 7121 | 7167 | 7001 | 6820 | 6624 | 6631 |
| 19 | **4124.01** | 5984 | 5721 | 5583 | 5607 | 5594 | 5568 | 4937 | 4755 | 4642 |
| 20 | **5651.8** | 7142 | 7018 | 6911 | 6901 | 6886 | 6893 | 6764 | 6734 | 6458 |
| 21 | **7891.56** | 9351 | 9274 | 9031 | 9022 | 9146 | 9048 | 8801 | 8739 | 8672 |
| 22 | **8172.09** | 9724 | 9328 | 9134 | 9271 | 9210 | 9161 | 9075 | 9013 | 8866 |
| 23 | **8133** | 10131 | 9749 | 9536 | 9561 | 9537 | 9548 | 8973 | 8827 | 8713 |
| 24 | **4426.97** | 5635 | 5272 | 5013 | 5047 | 5003 | 5043 | 4822 | 4780 | 4700 |
| 25 | **10456** | 13561 | 12732 | 12418 | 12502 | 12428 | 12521 | 11927 | 11801 | 11681 |
| 26 | **10661** | 14126 | 13254 | 13170 | 13211 | 13129 | 13146 | 12473 | 12276 | 12111 |
| 27 | **5440** | 7248 | 6773 | 6527 | 6504 | 6511 | 6516 | 6014 | 5789 | 5794 |
| 28 | **19506** | 26457 | 24115 | 23874 | 23689 | 23762 | 23706 | 22976 | 22955 | 22921 |
| 29 | **21019** | 24161 | 23572 | 22745 | 22512 | 22501 | 22629 | 22352 | 22147 | 22012 |
| 30 | **14527** | 17286 | 17128 | 16610 | 16268 | 16212 | 16324 | 16137 | 16019 | 15987 |
| 31 | **19009** | 24107 | 23741 | 23167 | 23214 | 23152 | 23160 | 22761 | 22518 | 22104 |
| 32 | **18203.11** | 26537 | 25743 | 24247 | 24342 | 24267 | 24212 | 21322 | 20790 | 20687 |
| 33 | **19006.76** | 27111 | 25732 | 23854 | 23381 | 23639 | 23417 | 22180 | 22015 | 21813 |
| 34 | **12896.01** | 18376 | 17527 | 16632 | 16511 | 16438 | 16421 | 15133 | 14708 | 14662 |
| 35 | **8800.52** | 11614 | 11626 | 10767 | 10704 | 10738 | 10701 | 9372 | 9169 | 9014 |
| 36 | **4365.94** | 6738 | 6172 | 5860 | 5217 | 5229 | 5247 | 4934 | 4551 | 4571 |

Table 4: The $p$-values of Wilcoxon statistical test for MA versus GA, $GA_{FT}$, $GA_{CX_1}$, $GA_{CX_2}$, $GA_{CX_3}$, $GA_{CX_4}$, $GA_{LS_1}$, $GA_{LS_2}$ and $GA_{LS_3}$

| **MA versus** **Inst** | **GA** | $GA_{FT}$ | $GA_{CX_1}$ | $GA_{CX_2}$ | $GA_{CX_3}$ | $GA_{CX_4}$ | $GA_{LS_1}$ | $GA_{LS_2}$ | $GA_{LS_3}$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | + | + | + | + | + | + | + | + | + |
| 2 | + | + | + | + | + | + | + | + | + |
| 3 | + | + | + | + | + | + | + | + | + |
| 4 | + | + | + | + | + | + | + | + | + |
| 5 | + | + | + | + | + | + | + | + | + |
| 6 | + | + | + | + | + | + | + | + | + |
| 7 | + | + | + | + | + | + | + | + | + |
| 8 | + | + | + | + | + | + | + | + | + |
| 9 | + | + | + | + | + | + | + | + | + |
| 10 | + | + | + | + | + | + | + | + | + |
| 11 | + | + | + | + | + | + | + | + | + |
| 12 | + | + | + | + | + | + | + | + | + |
| 13 | + | + | + | + | + | + | + | + | + |
| 14 | + | + | + | + | + | + | + | + | + |
| 15 | + | + | + | + | + | + | + | + | + |
| 16 | + | + | + | + | + | + | + | + | + |
| 17 | + | + | + | + | + | + | + | + | + |
| 18 | + | + | + | + | + | + | + | + | + |
| 19 | + | + | + | + | + | + | + | + | + |
| 20 | + | + | + | + | + | + | + | + | + |
| 21 | + | + | + | + | + | + | + | + | + |
| 22 | + | + | + | + | + | + | + | + | + |
| 23 | + | + | + | + | + | + | + | + | + |
| 24 | + | + | + | + | + | + | + | + | + |
| 25 | + | + | + | + | + | + | + | + | + |
| 26 | + | + | + | + | + | + | + | + | + |
| 27 | + | + | + | + | + | + | + | + | + |
| 28 | + | + | + | + | + | + | + | + | + |
| 29 | + | + | + | + | + | + | + | + | + |
| 30 | + | + | + | + | + | + | + | + | + |
| 31 | + | + | + | + | + | + | + | + | + |
| 32 | + | + | + | + | + | + | + | + | + |
| 33 | + | + | + | + | + | + | + | + | + |
| 34 | + | + | + | + | + | + | + | + | + |
| 35 | + | + | + | + | + | + | + | + | + |
| 36 | + | + | + | + | + | + | + | + | + |

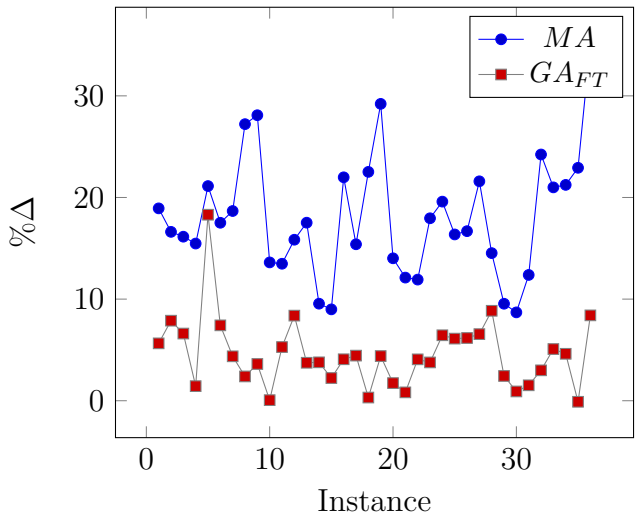


Figure 6: The percentage improvement of $GA_{FT}$ and MA from GA results

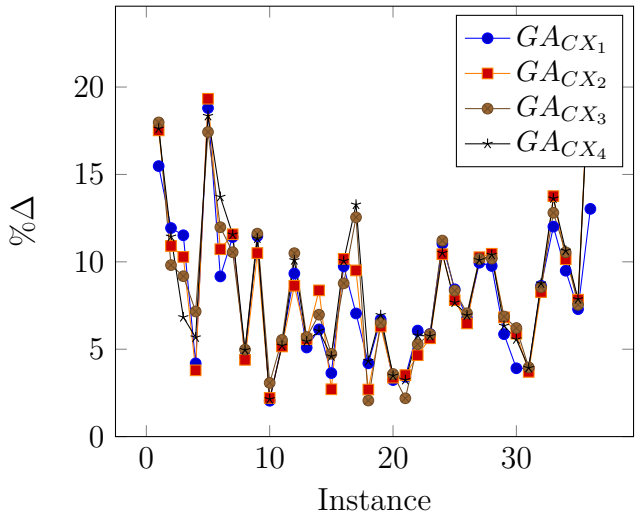


Figure 7: The percentage improvement of $GA_{CX_1}$, $GA_{CX_2}$, $GA_{CX_3}$ and $GA_{CX_4}$ from GA results

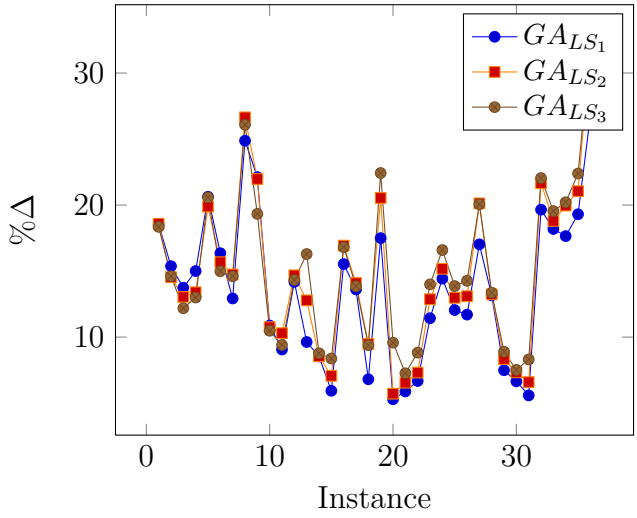


Figure 8: The percentage improvement of $GA_{LS_1}$, $GA_{LS_2}$ and $GA_{LS_3}$ from GA results

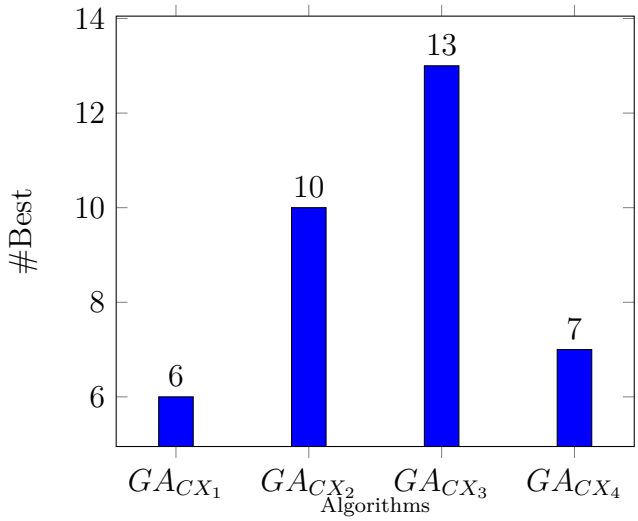


Figure 9: The number of best solutions obtained by $GA_{CX_1}$, $GA_{CX_2}$, $GA_{CX_3}$ and $GA_{CX_4}$

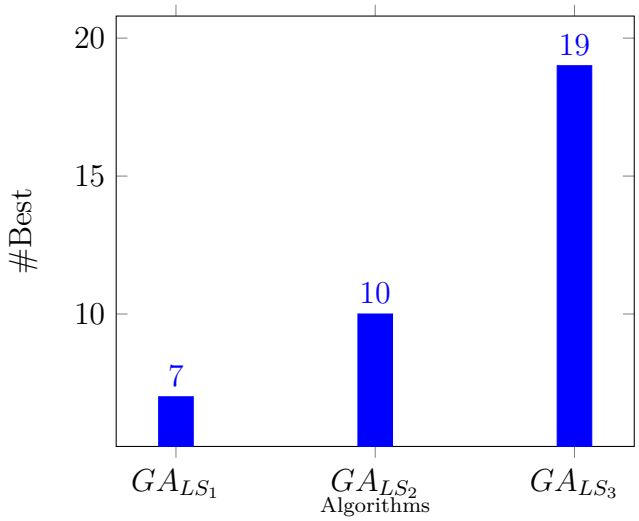


Figure 10: The number of best solutions obtained by $GA_{LS_1}$, $GA_{LS_2}$ and $GA_{LS_3}$

*5.2. Comparison with state-of-the-art algorithms*

This section is dedicated to an evaluation of MA results against those obtained by the state-of-the-art algorithms. Only three algorithms have been tested in the literature:

- SA_HLS: simulated annealing with heuristic local search [15].
- SA: simulated annealing algorithm [15].
- HSA: a hybrid of artificial bee colony and artificial immune system algorithm [32].

Table 5 presents the comparative results of MA and the three reference algorithms (SA_HLS, SA, HSA) In the table, we report the best obtained results for each class and the computational time. The best results achieved by the compared algorithms are highlighted in boldfont. From Table 5, it can be observed that the proposed MA achieves the best results for 32 out of the 36 tested instances.

Table 5 also discloses that MA is worse than HSA on four instance. Nevertheless, the percentage deviation for the other instances is relatively small indicating that the results of MA are very close to HSA. More importantly, the computational times of MA across all instances are smaller than SA,

HSA and SA_HLS. These outcomes provide evidence of the effectiveness of the proposed MA in dealing with 2L-HFVRP.

To verify the above positive result, we have applied the multiple statistical comparisons procedure described in [6] to compare the results of our approach against other algorithms. We first conducted Friedman and Iman–Davemport statistical tests (with a significance level of 0.05) to properly control the FWER and to detect if there is a significant differences between the compared algorithms. The obtained $p$-values ($p$-value=0.000) are below than 0.05 indicating that there is a significant difference between the compared algorithms (MA, SA_HLS, SA and HSA). As a result, we applied the Friedman test to calculate the average ranking of each algorithm as shown in Table 6. As can be seen from Table in 6, MA is ranked first with HSA, SA_HLS and SA following, in the order given. Consequently, MA is the best performing algorithm and it will be used as the control algorithm for the a post-hoc statistical tests (Holm and Hochberg statistical tests). The adjusted $p$-values of the post-hoc statistical tests for MA versus SA_HLS, SA and HSA are presented in Table 7 demonstrate that MA outperforms all other algorithms (SA_HLS, SA and HSA) with a critical level of 0.05 (all adjusted p-values are less than 0.05).

The numerical results presented throughout this work reveal that the proposed MA produced favourable results compared to other algorithms in the literature. Statistical tests also support these results. The good results, we believe, can be attributed to the following factors:

- The ability of the proposed MA in selecting, for each instance, different evolutionary operators and local search algorithms. By selecting, for each instance, different sequence of operators and local search algorithms, the proposed MA can deal with changes that might occur during the search, escaping from the local optima and exploring different areas in the problem-solution search space.

- The ability of the proposed multi-parent crossover operators in handling the diversity and quality of the search space as well as exploiting the knowledge obtained in previous generations to guide the search process. By utilising previous experience, MA can produce high quality and diverse offspring for the next generation.

- The ability of the proposed time-varying fitness function in guiding the evolutionary process to explore quality and diversity areas. By using

Table 5: The results of MA compared to the state of the art algorithms

| Inst | MA | | SA_HLS | | SA | | HSA | |
|---|---|---|---|---|---|---|---|---|
| | **Best** | **Time** | **Best** | **Time** | **Best** | **Time** | **Best** | **Time** |
| 1 | **587.2** | **13.1** | 600.8 | 29.9 | 668.4 | 26.48 | 598.02 | 15.26 |
| 2 | **676.4** | **11.01** | 699.2 | 32.9 | 755.4 | 27.56 | 694.64 | 11.78 |
| 3 | **758.5** | **16.23** | 770.1 | 33.8 | 856.7 | 49.55 | 764.12 | 21.19 |
| 4 | **677.7** | **17.22** | 698.2 | 30 | 765.8 | 54.02 | 696.82 | 18.92 |
| 5 | 783.4 | **18.07** | 786.8 | 27.7 | 902.1 | 68.51 | **765.67** | 75.87 |
| 6 | 826.2 | **33.2** | 831.3 | 42.8 | 905.3 | 59.54 | **825.68** | 35.19 |
| 7 | **5323.04** | **18.6** | 5630 | 31.1 | 6780 | 132.14 | 5344.54 | 63.46 |
| 8 | **5364.01** | **21.2** | 5603 | 30.2 | 7037 | 114.14 | 5398.68 | 45.72 |
| 9 | **1032** | **31.11** | 1036 | 58.8 | 1184 | 64.45 | 1035.62 | 34.45 |
| 10 | **6891.14** | **29.1** | 7625 | 43.3 | 8398 | 264.91 | 6920.34 | 107.84 |
| 11 | **7784** | **38.22** | 8330 | 52.6 | 9179 | 277.96 | 7792.35 | 127.35 |
| 12 | **1666.86** | **10.36** | 1681 | 167 | 1709 | 37.74 | 1681.07 | 12.26 |
| 13 | 25257.08 | **66.01** | 25979 | 70 | 31078 | 403.16 | **25252.24** | 145.3 |
| 14 | 10668 | **61.44** | 10869 | 78.1 | 11864 | 138.8 | **10639.67** | 159.55 |
| 15 | **11083.63** | **73.14** | 11490 | 91.5 | 12355 | 230.36 | 11091.77 | 162.63 |
| 16 | **1280.22** | **19.04** | 1292 | 111 | 1433 | 59.66 | 1289.57 | 19.89 |
| 17 | **1754.84** | **14.7** | 1777 | 191 | 1985 | 57.94 | 1773.58 | 16.7 |
| 18 | **5400** | **77.34** | 5676 | 88.8 | 7127 | 674.9 | 5410.67 | 288.27 |
| 19 | **4124.01** | **112.11** | 4242 | 180 | 5823 | 937.74 | 4124.33 | 175.53 |
| 20 | **5651.8** | **146.26** | 6153 | 175 | 8736 | 1722.4 | 5654.04 | 697.14 |
| 21 | **7891.56** | **200.03** | 8221 | 292 | 13922 | 2005.2 | 7904.08 | 265.8 |
| 22 | **8172.09** | **310.76** | 8574 | 317 | 14508 | 1972.6 | 8176.26 | 432.11 |
| 23 | **8133** | **297.41** | 8317 | 358 | 14579 | 1884.2 | 8147.26 | 661.39 |
| 24 | **4426.97** | **283.03** | 4548 | 289 | 6399 | 1156.4 | 4430.17 | 641.61 |
| 25 | **10456** | **379.8** | 11368 | 474 | 20788 | 3324.2 | 10480.26 | 425.09 |
| 26 | **10661** | **294.58** | 11782 | 363 | 19654 | 2679.5 | 10662.11 | 777.78 |
| 27 | **5440** | **277.08** | 5695 | 282 | 9010 | 2622.4 | 5463.73 | 650.28 |
| 28 | **19506** | **501** | 22611 | 614 | 39332 | 4547.4 | 19559.36 | 816.64 |
| 29 | **21019** | **434.26** | 21876 | 495 | 34554 | 6134.2 | 21037.22 | 1523.89 |
| 30 | **14527** | **781.07** | 15793 | 812 | 34011 | 6133.4 | 14612.82 | 1226.48 |
| 31 | **19009** | **992.73** | 21126 | 1115 | 47085 | 10052 | 19083.75 | 1522.45 |
| 32 | **18203.11** | **887.05** | 20111 | 969 | 47403 | 11131 | 18204.17 | 2006.68 |
| 33 | **19006.76** | **760.14** | 21420 | 858 | 50534 | 12506 | 19010.64 | 1427.65 |
| 34 | **12896.01** | **1127.32** | 14485 | 1596 | 24772 | 12471 | 12960.8 | 2232.99 |
| 35 | **8800.52** | **1119.05** | 8962 | 1214 | 12053 | 15793 | 8802.64 | 3109.13 |
| 36 | **4365.94** | **985.25** | 4386 | 1120 | 6120 | 15381 | 4383.01 | 2909.05 |

Table 6: Average Rankings of the algorithms (Friedman test)

| Algorithm | Ranking |
|---|---|
| MA | 1.1111 |
| SA_HLS | 2.9722 |
| SA | 4 |
| HSA | 1.9167 |

Table 7: Adjusted $p$-values obtained through the application of the post hoc methods: Holm and Hochberg statistical tests

| i | algorithm | unadjusted $p$ | $p_{Holm}$ | $p_{Hochberg}$ |
|---|---|---|---|---|
| 1 | SA_HLS | 0.000 | 0.000 | 0.000 |
| 2 | SA | 0.000 | 0.000 | 0.000 |
| 3 | HSA | 0.008113 | 0.008113 | 0.008113 |

the time-varying fitness and diversity calculation, the proposed MA focuses on exploration in the early stages and then gradually decreases so it can effectively explore new areas in the solution search space and escape from the basin of attraction points..

- Since most of the state-of-the-art algorithms heavily rely on a local search algorithm, the proposed MA adaptively combines the strengths of several local search algorithms in a unified framework..

## 6. Conclusion

This work proposed a two-stage method to solve the heterogeneous fleet vehicle routing problem with the two-dimensional loading constraints problems. In the first stage, we have proposed an adaptive memetic approach to handle the routing process. The proposed memetic approach integrates various effective and complimentary elements to generate high quality solutions. Four different multi-parent crossover operators that use diversity and quality were proposed to evolve a new population of solutions. To accelerate the search process, three different local search algorithms are adaptively applied to further improve the evolved solutions. We propose a time varying fitness function to avoid prematurity. We proposed an adaptive quality-and-diversity mechanism to control the application of memetic operators and local search algorithms. In the second stage, five different heuristics were adopted

and hybridised to perform the packing process. The experimental results using the existing benchmark instances demonstrate that the proposed algorithm can obtain very good results, if not better on many instances, when compared to the state-of-the-art algorithms.

## References


[1] Azad, A. S., Islam, M. M., and Chakraborty, S. (2017). A heuristic initialized stochastic memetic algorithm for mdpvrp with interdependent depot operations. *IEEE Transactions on Cybernetics*, PP(99):1–14.

[2] Bräysy, O. and Gendreau, M. (2005). Vehicle routing problem with time windows, part i: Route construction and local search algorithms. *Transportation science*, 39(1):104–118.

[3] Choong, S. S., Wong, L.-P., and Lim, C. P. (2019). An artificial bee colony algorithm with a modified choice function for the traveling salesman problem. *Swarm and evolutionary computation*, 44:622–635.

[4] Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management science*, 6(1):80–91.

[5] Decerle, J., Grunder, O., El Hassani, A. H., and Barakat, O. (2019). A memetic algorithm for multi-objective optimization of the home health care problem. *Swarm and evolutionary computation*, 44:712–727.

[6] Derrac, J., García, S., Molina, D., and Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18.

[7] Eiben, A. E. (2003). Multiparent recombination in evolutionary computing. *Advances in evolutionary computing*, pages 175–192.

[8] Fialho, Á., Da Costa, L., Schoenauer, M., and Sebag, M. (2010). Analyzing bandit-based adaptive operator selection mechanisms. *Annals of Mathematics and Artificial Intelligence*, 60(1-2):25–64.

[9] García, J., Crawford, B., Soto, R., and Astorga, G. (2019). A clustering algorithm applied to the binarization of swarm intelligence continuous metaheuristics. *Swarm and evolutionary computation*, 44:646–664.

[10] Iori, M., Salazar-González, J.-J., and Vigo, D. (2007). An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, 41(2):253–264.

[11] Karafotias, G., Hoogendoorn, M., and Eiben, A. E. (2015). Parameter control in evolutionary algorithms: Trends and challenges. *IEEE Transactions on Evolutionary Computation*, 19(2):167–187.

[12] Khan, I. and Maiti, M. K. (2019). A swap sequence based artificial bee colony algorithm for traveling salesman problem. *Swarm and evolutionary computation*, 44:428–438.

[13] Koç, Ç., Bektaş, T., Jabali, O., and Laporte, G. (2015). A hybrid evolutionary algorithm for heterogeneous fleet vehicle routing problems with time windows. *Computers & Operations Research*, 64:11–27.

[14] Kubiak, M. (2007). Distance measures and fitness-distance analysis for the capacitated vehicle routing problem. In *Metaheuristics*, pages 345–364. Springer.

[15] Leung, S. C., Zhang, Z., Zhang, D., Hua, X., and Lim, M. K. (2013). A meta-heuristic algorithm for heterogeneous fleet vehicle routing problems with two-dimensional loading constraints. *European Journal of Operational Research*, 225(2):199–210.

[16] Ong, Y. S. and Keane, A. J. (2004). Meta-lamarckian learning in memetic algorithms. *Evolutionary Computation, IEEE Transactions on*, 8(2):99–110.

[17] Pavai, G. and Geetha, T. V. (2016). A survey on crossover operators. *ACM Comput. Surv.*, 49(4):72:1–72:43.

[18] Pollaris, H., Braekers, K., Caris, A., Janssens, G. K., and Limbourg, S. (2015). Vehicle routing problems with loading constraints: state-of-the-art and future directions. *OR Spectrum*, 37(2):297–330.

[19] Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002.

[20] Sabar, N. R., Abawajy, J., and Yearwood, J. (2017a). Heterogeneous cooperative co-evolution memetic differential evolution algorithm for big data optimization problems. *IEEE Transactions on Evolutionary Computation*, 21(2):315–327.

[21] Sabar, N. R. and Aleti, A. (2017). An adaptive memetic algorithm for the architecture optimisation problem. In *Australasian Conference on Artificial Life and Computational Intelligence*, pages 254–265. Springer.

[22] Sabar, N. R. and Ayob, M. (2009). Examination timetabling using scatter search hyper-heuristic. In *2009 2nd Conference on Data Mining and Optimization*, pages 127–131. IEEE.

[23] Sabar, N. R., Ayob, M., Kendall, G., and Qu, R. (2013). Grammatical evolution hyper-heuristic for combinatorial optimization problems. *IEEE Transactions on Evolutionary Computation*, 17(6):840–861.

[24] Sabar, N. R., Ayob, M., Kendall, G., and Qu, R. (2014a). Automatic design of a hyper-heuristic framework with gene expression programming for combinatorial optimization problems. *IEEE Transactions on Evolutionary Computation*, 19(3):309–325.

[25] Sabar, N. R., Ayob, M., Kendall, G., and Qu, R. (2014b). A dynamic multiarmed bandit-gene expression programming hyper-heuristic for combinatorial optimization problems. *IEEE transactions on cybernetics*, 45(2):217–228.

[26] Sabar, N. R., Bhaskar, A., Chung, E., Turky, A., and Song, A. (2019). A self-adaptive evolutionary algorithm for dynamic vehicle routing problems with traffic congestion. *Swarm and evolutionary computation*, 44:1018–1027.

[27] Sabar, N. R., Chung, E., Tsubota, T., de Almeida, P. E. M., et al. (2017b). A memetic algorithm for real world multi-intersection traffic signal optimisation problems. *Engineering Applications of Artificial Intelligence*, 63:45–53.

[28] Sabar, N. R., Song, A., and Zhang, M. (2016). A variable local search based memetic algorithm for the load balancing problem in cloud computing. In *European Conference on the Applications of Evolutionary Computation*, pages 267–282. Springer.

[29] Sadhu, A. K., Konar, A., Bhattacharjee, T., and Das, S. (2018). Synergism of firefly algorithm and q-learning for robot arm path planning. *Swarm and Evolutionary Computation*, 43:50–68.

[30] Talbi, E.-G. (2009). *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons.

[31] Toth, P. and Vigo, D. (2014). *Vehicle Routing: Problems, Methods, and Applications*, volume 18. SIAM.

[32] Zhang, D., Dong, R., Si, Y.-W., Ye, F., and Cai, Q. (2018). A hybrid swarm algorithm based on abc and ais for 2l-hfcvrp. *Applied Soft Computing*, 64:468–479.

[33] Zhang, G. and Li, Y. (2016). A memetic algorithm for global optimization of multimodal nonseparable problems. *IEEE Transactions on Cybernetics*, 46(6):1375–1387.