Knowledge-Based Systems 230 (2021) 107378

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

An improved learnable evolution model for solving multi-objective vehicle routing problem with stochastic demand

Yunyun Niu [a], Detian Kong [a], Rong Wen [b], Zhiguang Cao [b], Jianhua Xiao [c,*]

[a] School of Information Engineering, China University of Geosciences in Beijing, Beijing 100083, China
[b] Singapore Institute of Manufacturing Technology, Singapore 138634, Singapore
[c] The Research Center of Logistics, Nankai University, Tianjin 300071, China

ARTICLE INFO

Article history:
Received 8 April 2021
Received in revised form 11 July 2021
Accepted 9 August 2021
Available online 14 August 2021

Keywords:
Vehicle routing problems
Stochastic demand
Learnable evolution model
Multi-objective evolutionary algorithm

ABSTRACT

The multi-objective vehicle routing problem with stochastic demand (MO-VRPSD) is much harder to tackle than other traditional vehicle routing problems (VRPs), due to the uncertainty in customer demands and potentially conflicted objectives. In this paper, we present an improved multi-objective learnable evolution model (IMOLEM) to solve MO-VRPSD with three objectives of travel distance, driver remuneration and number of vehicles. In our method, a machine learning algorithm, i.e., decision tree, is exploited to help find and guide the desirable direction of evolution process. To cope with the key issue of "route failure" caused due to stochastic customer demands, we propose a novel chromosome representation based on priority with bubbles. Moreover, an efficient nondominated sort using a sequential search strategy (ENS-SS) in conjunction with some heuristic operations are leveraged to handle the multi-objective property of the problem. Our algorithm is evaluated on the instances of modified Solomon VRP benchmark. Experimental results show that the proposed IMOLEM is capable to find better Pareto front of solutions and also deliver superior performance to other evolutionary algorithms.

1. Introduction

The vehicle routing problem (VRP) is a classic combinatorial optimization problem in operations research and computer science, with wide applications in the field of logistics. In a typical VRP, the vehicles at a depot perform delivery (or pickup) of goods to (or from) customers at various locations. In general, the depot and customers are characterized by coordinates, and each customer also has certain demand. The vehicles with capacity constraints, depart from the depot and visit the customers sequentially, then return to the depot. The goal of VRP is usually to determine an entire route which can satisfy all the customers demands while complying with the operational constraints, in ways that the total costs are minimized [1].

Different from the typical VRP, the stochastic VRP (SVRP) is usually characterized by some random parameters [2–5]. The common ones include stochastic demand [6], stochastic customer [7–9], and stochastic travel and service time [10,11]. Compared with the other two, stochastic demand is relatively ubiquitous and has more real-world applications such as oil delivery [12], trash collecting, sludge transporting [13], and the cash collecting of bank [14]. Therefore, we emphasize on the VRP with stochastic demands (VRPSD). Actually, addressing the VRPSD is non-trivial as the stochastic demands may make the problem more intricate. In VRPSD, the actual demand of each customer is unknown until the vehicle arrives at the location of customer. Since the vehicle route is usually planned in advance, it may cause "route failure" when the vehicle arrives at the customer location and found that the goods are insufficient [15,16]. In that case, the vehicle has to return to the depot for replenishment and visit the customer again, which renders it much harder to optimize the cost. However, what makes it more challenging is that, VRPSD is usually investigated with multiple objectives, such as optimizing the travel distance, driver remuneration and number of vehicles. Therefore, in this paper, we aim to tackle the multi-objective VRPSD (MO-VRPSD) with the above three objectives.

Many exact methods and heuristic (or approximate) methods have been presented for solving VRPSD or MO-VRPSD. However, the exact methods always become far less efficient as the problem scales up. So in reality, researchers often rely on approximate methods to solve the problem by finding a satisfactory solution in reasonably short computation time. On the one hand, the VRP could be interpreted as a sequential decision making problem with Markov decision process (MDP) properties, where both approximate dynamic programming (ADP) approach and reinforcement learning (RL) approach have been proposed to solve the respective VRPs. Nicola et al. [17] present a re-optimization

* Corresponding author.
E-mail address: jhxiao@nankai.edu.cn (J. Xiao).

https://doi.org/10.1016/j.knosys.2021.107378
0950-7051/

approach for solving the VRPSD, where they leveraged a finite-horizon MDP formulation for single-vehicle VRPSD and proposed a heuristic algorithm for the MDP. Clara et al. [18] examined ADP algorithms for the single-vehicle VRPSD from a dynamic or re-optimization perspective by extending the rollout algorithm. Nazari et al. [19] developed a framework with the capability to solve a wide variety of combinatorial optimization problems using RL and demonstrated how it can be applied to solve the single-vehicle VRP, including single-vehicle VRPSD. However, the VRPSD models based on Markov decision process is often limited to a single vehicle, where they focused on optimizing their goals with one vehicle. Moreover, rare works have been investigated to exploit the nature of MDP for solving multi-objective VRPSD variants. Most of them only considered one objective such as travel distance [20].

On the other hand, a number of classic heuristic algorithms also have been exploited to cope with VRPSD or MO-VRPSD, such as simulated annealing algorithm [21], tabu search algorithm [22], genetic algorithm based approaches [23], particle swam optimization algorithm [24] and artificial immune algorithm [25]. Due to the better generalization capability, multi-objective evolutionary algorithms have been widely applied to solve various problems including manufacturing scheduling problems, feature selection problems, engineering optimization problems and so on [26,27]. Particularly, they could be combined with the above heuristic algorithm (such as particle swarm optimization [28,29]), and have achieved desirable results [30,31]. Tan et al. presented a multi-objective evolutionary algorithm to engender desirable trade-off solutions [32]. Gee et al. proposed a decomposition-based multi-objective evolutionary algorithm [33]. Although demonstrating certain favourable performance, those evolutionary computation algorithms are essentially Darwinian-type. In this context, they employ genetic operations to simulate creature evolution, where the search process is semi-blind and may cause degeneration of the solution quality and waste of computational time. Unlike the Darwinian-type evolutionary algorithms, learnable evolution model (LEM) is a different yet more attractive class of evolutionary algorithm, which leverages machine learning to guide the evolution direction [34]. Particularly, LEM learns the criteria of superior individuals during the evolutionary process, based on which it will generate new population. In doing so, LEM is supposed to avoid the semi-random search process and reduce the computational time considerably. Among this line of works, Moradi et al. proposed a multi-objective discrete learnable evolution model to handle the vehicle routing problem with time windows (VRPTW), and demonstrated superior results to others [35]. Unfortunately, this algorithm cannot directly solve the MO-VRPSD due to the inferior representation of the chromosome. In specific, its representation is unable to handle the "route failure" in the MO-VRPSD, which tends to break the map between chromosome representation and routing scheme. To address this key issue, in this paper, we propose an improved LEM model for solving the MO-VRPSD by mainly designing a novel chromosome representation, i.e., IMOLEM. The major contributions are summarized as follows.

- We present a decision tree based LEM to solve the MO-VRPSD, which helps generate desirable populations according to the knowledge learned from past searching process. This learning mechanism avoids the semi-blind search and reduces the number of iterations remarkably. Several heuristic operators are also developed to produce better initial population and increase the diversity during the evolutionary process.
- We propose a new approach to encode and decode the route and chromosome so that it is able to effectively deal with "route failure" caused by uncertain customer demands. To ensure the uniqueness of chromosome representation, some bubbles are inserted into the priority sequence. Note that the proposed representation scheme plays a key role in applying LEM to solve the MO-VRPSD.
- We leverage a robust multi-objective optimization algorithm, i.e., efficient nondominated sort using a sequential search strategy (ENS-SS) [36], to handle the conflicting objectives in the MO-VRPSD. With this algorithm, the diversity of the found solutions is enhanced which may lead to a better solution set.

2. Problem formulation

In our MO-VRPSD model, a set of customers at various locations have certain demands of goods. A fleet of vehicles depart from the central depot to deliver the goods to the customers, and return to the depot finally. In specific, the set $V = \{v_0, v_1, v_2, \ldots, v_N\}$ represents the depot (v_0) and N customers to be served. Each customer v_i has a coordinate (x_i, y_i) (the coordinate for the depot is (x_0, y_0)), and a stochastic demand D_i following a standard normal distribution, whose mean is μ_i and variance is σ_i^2. The actual demand d_i is unknown until the vehicle arrives at the customer location. Each customer has a service time s_i during which the vehicle needs to stay at the customer to complete the delivery service. The vehicles are assumed to be homogeneous with a capacity C.

The sub-route for a vehicle always starts at the central depot, and gets through a set of customers, then ends at the central depot. Accordingly, the sub-route could be represented as a sequence of customers $R(r) = [v_0, n_1(r), n_2(r), \ldots, n_k(r), v_0]$, where k is the number of customers to serve on this sub-route, and $n_k(r) \in (V - \{v_0\})$ with $n_1(r) \neq n_2(r) \neq \ldots \neq n_k(r)$. The travel distance c_{ij} between any two points (i.e., customer or depot) v_i and v_j can be calculated as follows

$$c_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}. \tag{1}$$

In Eq. (1), we assume that the travel time is proportional to the distance,[1] and set the proportion coefficient as 1 to simplify the computation. So c_{ij} also refers to the travel time between v_i and v_j. A route *failure* may occur when the remaining goods in the vehicle is not adequate to serve the next customer. For sub-route $R(r) = [v_0, n_1(r), n_2(r), \ldots, n_f(r), \ldots, n_k(r), v_0]$, the route failure will occur at customer $n_f(r)$ as long as $\sum_{i=1}^{f} d_{n_i(r)} \geqslant C$. To ensure the feasibility of solutions, a simple recourse policy is employed. In particular, if the vehicle arrives at customer $n_f(r)$ and finds that the remaining goods are less than $d_{n_f(r)}$ ($\sum_{i=1}^{f-1} d_{n_i(r)} < C$ and $\sum_{i=1}^{f} d_{n_i(r)} > C$), it will unload all the remaining goods (equivalent to $C - \sum_{i=1}^{f-1} d_{n_i(r)}$) at $n_f(r)$ and return to the depot for replenishment. Then, it will turn back to customer $n_f(r)$ and complete the remaining service (equivalent to $d_f - C + \sum_{i=1}^{f-1} d_{n_i(r)}$). For the case that after the vehicle services customer $n_f(r)(f < k)$ and finds the remaining goods is 0 ($\sum_{i=1}^{f} d_{n_i(r)} = C$), it will return to the depot directly for replenishment and then head to customer $n_{f+1}(r)$. These replenishing actions will incur additional transportation cost, i.e., the additional travel distance $Q_d(r)$ and additional travel duration $Q_t(r)$, where $Q_d(r)$ includes the distance for the fro trip to depot, and $Q_t(r)$ includes replenishment time and the additional service time due to visiting a customer multiple times.

[1] Our method could also be easily extended to other spaces besides the Euclidean one.

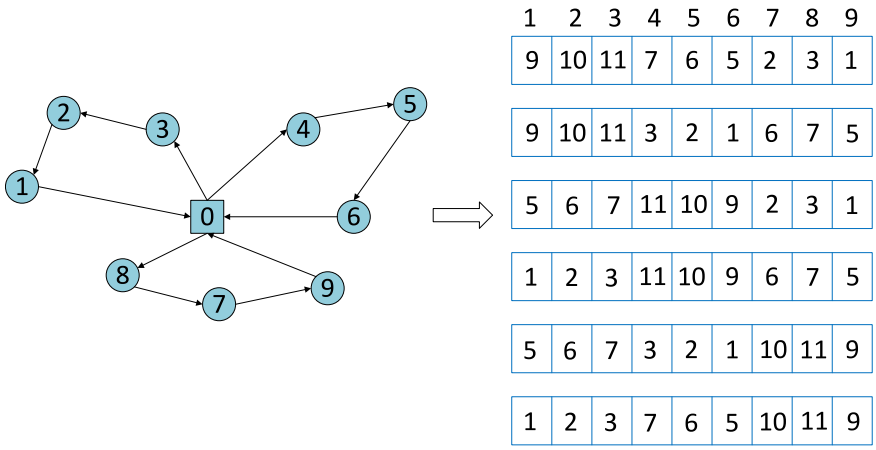


Fig. 1. An example of priority-based chromosome encoding.

Given the above statements, the total travel distance of sub-route $R(r)$ is expressed as follows

$$D(r) = c_{v_0,n_1(r)} + \sum_{i=1}^{k-1} c_{n_i(r),n_{i+1}(r)} + c_{n_k(r),v_0} + Q_d(r). \quad (2)$$

The total time duration of sub-route $R(r)$ is expressed as follows

$$T(r) = c_{v_0,n_1(r)} + \sum_{i=1}^{k-1} c_{n_i(r),n_{i+1}(r)} + c_{n_k(r),v_0} + \sum_{i=1}^{k} s_{n_i(r)} + Q_t(r). \quad (3)$$

We stipulate that each vehicle has a driver and he will get remuneration for the transportation, which is related to the time duration of the route. It is defined as follows

$$M(r) = \begin{cases} \frac{W}{B} T(r) m_1, & T(r) \leqslant B \\ W m_1 + (\frac{W}{B} T(r) - W) m_2, & T(r) > B, \end{cases} \quad (4)$$

where B is a given bound for working duration; if the driver is working overtime and will get more remuneration; if the total travel time of a vehicle is equal to B, then the driver is counted as working a normal duration, i.e., M hours; the remuneration rate for normal working duration is $\$m_1$ per hour and the rate for overtime is $\$m_2$ per hour ($m_2 > m_1$), respectively.

A feasible solution G is the entire route (a set of sub-routes) $G = \{R(r_1), R(r_2), \ldots, R(r_m)\}$ where m is the number of sub-routes, also refers to the number of vehicles used. The sub-routes R in G must include all customers in V. Accordingly, the goal of our MO-VRPSD is to find a solution $G = \{R(r_1), R(r_2), \ldots, R(r_m)\}$ that will minimize the three objectives, i.e., travel distance (D), driver remuneration (M) and number of vehicles (m) as follows

$$minimize \begin{cases} \sum_{i=1}^{m} D(r_i), \\ \sum_{i=1}^{m} M(r_i), \quad R(r_i) \in G, \\ m. \end{cases} \quad (5)$$

3. The improved multi-objective learnable evolution model

We propose an improved multi-objective learnable evolution model (IMOLEM) to solve the MO-VRPSD. The proposed model consists of three major components, i.e., *archive set generation*, *machine learning searching*, and *route simulation method* (RSM) [6], respectively. Particularly, the archive set generation maintains and updates the archive set based on a novel chromosome representation scheme and ENS-SS. The machine learning searching helps guide the direction of evolution via hypothesis generating and instantiating based on a decision tree in conjunction with various heuristic operators. RSM helps assign fitness to the respective solutions. Note that, in the archive set generator, (1) the new chromosome representation scheme based on priority with bubbles enables IMOLEM to handle the "route failure" in MO-VRPSD more efficiently; (2) unlike most of existing multi-objective optimization algorithms, ENS-SS only needs to compare with the ones that have already been assigned to the Pareto front, which avoids unnecessary comparisons and thus improves the computation efficiency. Accordingly, the resulting algorithm, i.e., IMOLEM is summarized in algorithm 1. Firstly, the customers information is read from the problem instances, which includes the coordinates of customers and depot, the mean and variance of the demand distribution, and the service time of each customer. It also reads the capacity of vehicle and the soft time constraint. An archive set is created to maintain the solutions and will be updated during the evolution process. Then, the initialization process is performed and the algorithm starts the evolution process, where it generates and instantiates the hypothesis, and then updates the archive set. The evolution process will repeat until the termination conditions is satisfied. In the following, we will detail the key procedures in our algorithm.

3.1. Archive set generation based on novel chromosome representation

3.1.1. Chromosome representation

Solution encoding plays a key role in applying the LEM to solve the MO-VRPSD. A chromosome sequence with integer genes is usually adopted to represent a route solution, whose length is N corresponding to the customers in a problem instance. The integer values of the gene in a chromosome sequence represents the index of the customer in V to be served. In conventional VRP, an implicit rule in the encoding and decoding scheme usually exists, which imposes that capacity and time window constraints should not be violated. This rule normally serves as the separator for different sub-routes. In our MO-VRPSD, there is only soft time constraint. Violating the capacity constraint is tolerable in the case of "route failure" since replenishing is allowed given that the actual demand of a customer is not known until an actual visit occurred. In this regard, a new chromosome encoding and decoding approach is designed in the archive set generation for solving the MO-VRPSD.

Specifically, we exploit an indirect fixed-length representation with bubbles to encode a route solution, where the customer indexes do not directly appear on the chromosome representation. Instead, the numbers on the gene represent the visiting priority of the customers, which act as the guiding information when they

Algorithm 1: IMOLEM

Input: *size*, max_{iter}
Output: Q

```
read customers information from the problem instances;
read capacity and time constraints from the problem instances;
P ← population from heuristic initialization;
Q ← empty list;
clf ← create new decision tree;
for i ← 1: max_iter do
    for customer in customers do
        generate actual demands to customer;
    end
    for individual in Q do
        perform RSE to individual;
    end
    Q is extended from P;
    Q ← Pareto sort and truncate to size;
    P ← Pareto sort of P;
    Hgroup ← high part of P;
    Lgroup ← low part of P;
    fit clf with H-group and L-group;
    P ← population from instantiating with clf;
    P is extended from first half of Q;
    for individual in last half of P do
        perform heuristic search operation to individual;
    end
end
return Q
```

are used to construct the route. In this case, the index of a gene on the chromosome represents the index of customer in V and its value denotes the visiting priority of the customer for constructing a route. To ensure unique chromosome representation, i.e., the decoding result of a chromosome exactly corresponds to one route solution, some bubbles are leveraged and inserted into the priority sequence.

To create a chromosome for the entire route solution, we need first to calculate the max priority. Assume that the number of customers is denoted by N_c and the number of routes in the solution is denoted by N_r, then $max\_priority = N_c + N_r - 1$. We then choose a sub-route from the solution and assign *max_priority* to the first customer in the sub-route, and set $max\_priority = (max\_priority - 1)$. Subsequently, we choose the second customer and assign the current *max_priority* to it. We continue the same procedure until the current sub-route ends. Before starting the next sub-route, we set $max\_priority = (max\_priority - 1)$, to indicate the bubble, so that it ensures the uniqueness of chromosome representation. Afterwards, we choose the next sub-route, and repeat the same procedure in the first sub-route until all the sub-routes in the chromosome have been processed. Since there is not a given prescribed order among sub-routes in the entire route, the number of possible chromosomes in a route solution would be $N_r!$ (the factorial of N_r). In Fig. 1, an example of the priority-based chromosome encoding is displayed, where the entire route with 3 sub-routes could construct 6 different chromosomes. However, one chromosome can only decode into one entire route.

Contrary to the encoding scheme, the decoding scheme is designed to transform a chromosome sequence into a route solution including a set of sub-routes. In a sub-route the vehicle always departs from the central depot, visits a set of customers and then returns to the central depot. The sub-route is usually built by first appending the depot v_0 as the starting point, where all the customers are unserved initially. Then we append the

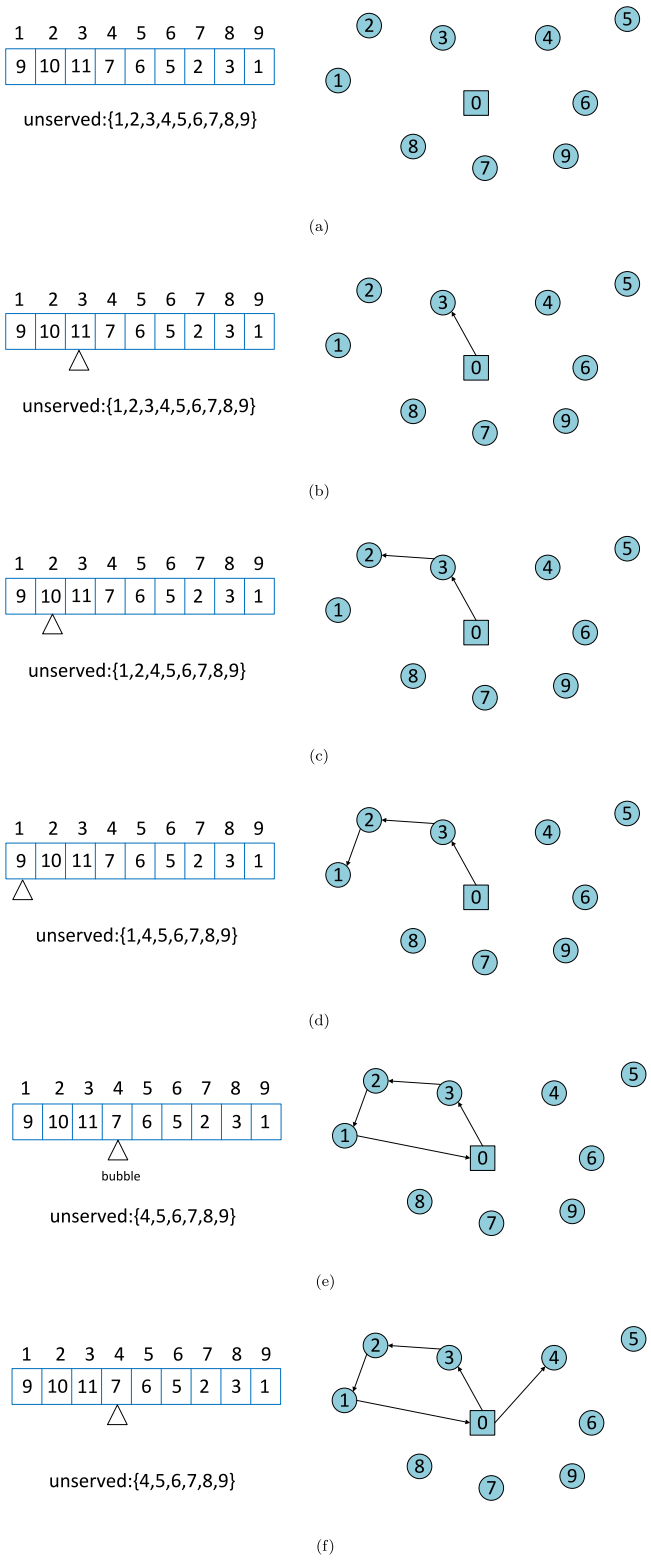


Fig. 2. An example of priority-based chromosome decoding.

customer with the highest priority from the unserved customer set to the sub-route, and also remove this customer from that set. Similarly, the next customer with the highest priority in the unserved customer set is selected. If priority difference is 2 for two consecutive customers, that means a bubble is met. In this case, the sub-route is ended and the vehicle should go back to

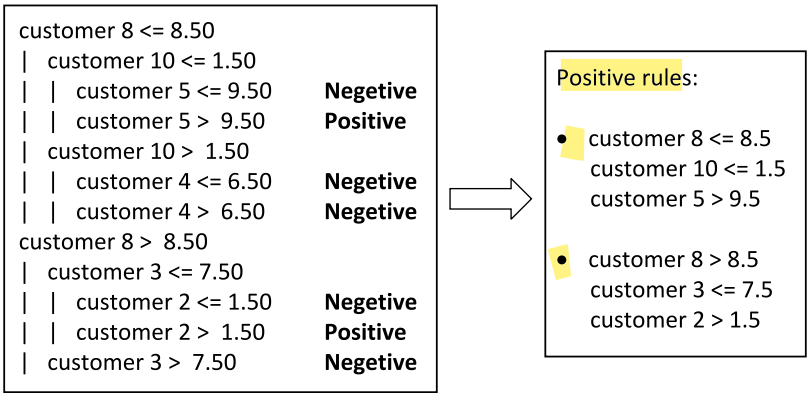


Fig. 3. The illustration of the decision tree algorithm.

the central depot, then the depot or v_0 is appended to the sub-route. Subsequently, another new sub-route will be constructed in a similar way. This process continues until all the customers are served. Fig. 2 presents an example instance of the priority-based chromosome decoding with bubbles. In specific, the customer with the highest priority is selected and appended to the sub-route at each step, and the unserved customer set is updated by removing the selected ones. The decoding result shows that the solution includes three sub-routes, i.e., R1: $0 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 0$, R2: $0 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 0$ and R3: $0 \rightarrow 8 \rightarrow 7 \rightarrow 9 \rightarrow 0$.

In the chromosome representation scheme, a sub-route appends v_0 as the end point when it meets a bubble in the chromosome sequence or all customers have been served. This decoding approach ensures the uniqueness for both chromosome and route solution, and the flexibility of chromosome instantiating. Consequently, this chromosome representation scheme in the archive set generation has the potential to lead the evolution to desirable directions and also enhance its performance.

3.1.2. Population initialization

The initial population is also important in the evolutionary algorithm. A well distributed initial population in the whole search space may significantly reduce the computation time of the multi-objective evolutionary algorithm and lead to a desirable approximation of the Pareto fronts. It is known that an initial population with a number of high-quality individuals can also help improve the convergence of algorithm. Conversely, a badly distributed initial population, i.e., the ones with completely random vehicle numbers and customer sequences, may consume prohibitively long computation time and also lead the evolutionary direction towards to local basins.

The maximum available vehicle number in our MO-VRPSD could be any integer value between 1 and customers number, given the replenishing policy and soft time constraint. However, it is unwise to choose a completely random value as the initial vehicle number. To build a high-quality initial population, we adopt an criterion as follows. In the first solution, the sum of the mean demand (we assume the mean is known in advance) of customers in every sub-route should not be larger than the vehicle capacity. In this case, if all customers actual demands equal to their mean values of demands, then route failure will not occur. Besides, the sum of the travelling time on every sub-route should not be longer than the soft time constraint, which means that the drivers will not work overtime and will get normal remuneration. The first chromosome will govern the maximum number of vehicles, and the ones in the chromosomes generated later will not exceed it. Following above criteria, a new solution is constructed by randomly picking a vehicle number from a reasonable range and then assign every sub-route to approximately

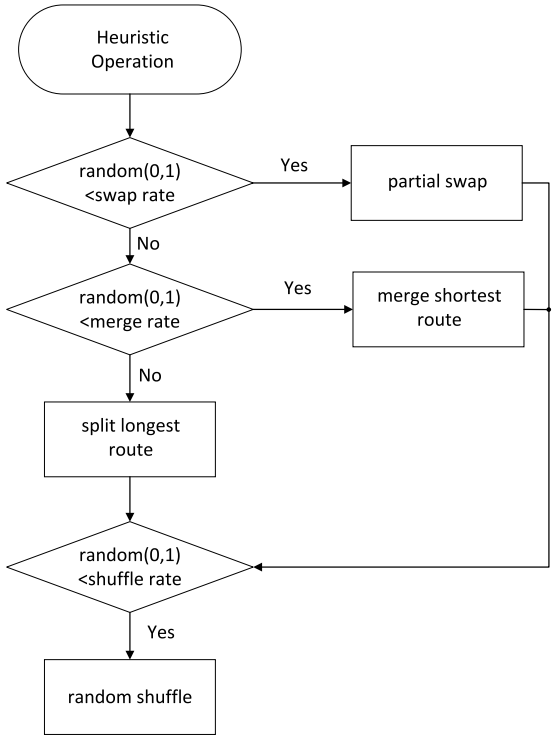


Fig. 4. Operation of heuristic search operation.

have identical number of customers. The sequence of customers in each sub-route is randomly assigned. The same procedure is repeated until a pre-defined number of chromosomes have been generated.

After the population initialization, the objective values related to a customer are not immediately known, then the RSM (see Section 3.3) is performed on each chromosome to obtain their objective values. An empty archive set is created to store the best chromosomes till present, whose size is equal to the one of the evolving population. All the chromosomes in the initial population are sorted in terms of their Pareto dominance relationship by the multi-objective optimization algorithm, i.e., ENS-SS, and then appended into the archive set. Note that, unlike most of existing multi-objective optimization algorithms, ENS-SS only needs to compare with the ones that have already been assigned to the Pareto front, which avoids unnecessary comparisons and thus improves the computation efficiency.

Table 1
Parameter settings.

Parameter	Values
Population size	100
No. of generations	200
H-group size	30%
L-group size	30%
Search rate	0.4
Swap rate	0.5
Split rate	0.5
Shuffle rate	0.3

3.2. Learnable evolutionary model with decision tree

After the population is initialized, the learnable evolution process using decision tree will start based on them. New generations will be produced iteratively through the hypothesis generating and instantiating process. Hypothesis generating refers to the learning process, where decision tree classifier will learn the knowledge from the current population and then generate hypothesis for the next hypothesis instantiating process. Hypothesis instantiating refers to the practice process, where new population will be generated through instantiating the hypothesis derived by the decision tree classifier, and served as new generation. Moreover, the new generation will offer knowledge for the decision tree to be updated. This procedure is repeated until the termination conditions are satisfied.

3.2.1. Hypothesis generating

The archive set is sorted by their Pareto dominance so the individuals in the head have better performance than the tail ones. Accordingly, a *H-group* (high performance group) and a *L-group* (low performance group) are formed by selecting the first *H%* and the last *L%* individuals from the archive set as the *positive* and *negative* training samples, respectively. The remaining individuals are not included since H-group and L-group need to be significantly different on objective values.

Intuitively, many machine learning algorithms could be leveraged in the LEM to guide the evolutionary direction. However, decision tree classifier is more suitable in our method. In particular, decision tree is able to yield a feasible region to each tree node, where the tree node refers to a customer index. In doing so, the decision tree is explicable for the prediction or decision it made. On the other hand, the results by general neural network or other machine learning algorithms are hard to explain so the instantiating procedure may only rely on its input and output, and treat the model as a black box. In contrast, the decision tree may exploit the classifier model itself by analysing its tree node so that the instantiating procedure would be more convenient.

Concretely, in our method, the decision tree classifier is employed to create hypothesizes to classify individuals between the H-group and L-group. The dual-type decision tree classifier treats the individuals in H-group as *positive* and individuals in L-group as *negative*, and then updates itself using the corresponding fitting algorithm according to the chromosomes of the individuals. The generated hypothesis is a set of rules which determine the chromosome with respective priority value ranges belongs to which group. The positive rules prefer to generate routes which may lead to high quality solutions and the negative rules are more likely to generate routes which may lead to low quality solutions. The rules for H-group and L-group derived by the decision tree classifier will be dynamically updated during the evolution process. The priority value ranges of the generated rules by the decision tree classifier would constantly converge to small ranges which may lead towards to the optimal MO-VRPSD solution. Fig. 3 illustrates the basic process of the decision tree classifier.

Table 2
Comparison of the IMOLEM and IMOLEM$_{_dtc}$.

		NV	TD	DR	HV	Spacing
C1	IMOLEM	8.20	3152.07	1960.47	0.05	37.62
	IMOLEM$_{_dtc}$	7.52	3922.86	2519.86	0.07	81.55
C2	IMOLEM	8.95	2880.87	1734.37	0.08	36.64
	IMOLEM$_{_dtc}$	14.58	3916.64	2067.12	0.07	43.42
R1	IMOLEM	8.20	2535.08	1949.00	0.07	27.03
	IMOLEM$_{_dtc}$	10.26	3314.36	2387.66	0.05	55.89
R2	IMOLEM	11.00	2545.12	1746.85	0.05	36.43
	IMOLEM$_{_dtc}$	12.88	3281.93	2193.90	0.05	40.15
RC1	IMOLEM	10.00	3527.05	2041.93	0.06	34.48
	IMOLEM$_{_dtc}$	11.58	4511.21	2577.39	0.07	46.83
RC2	IMOLEM	10.04	3198.93	1827.90	0.08	37.65
	IMOLEM$_{_dtc}$	12.25	4276.62	2365.25	0.08	61.61

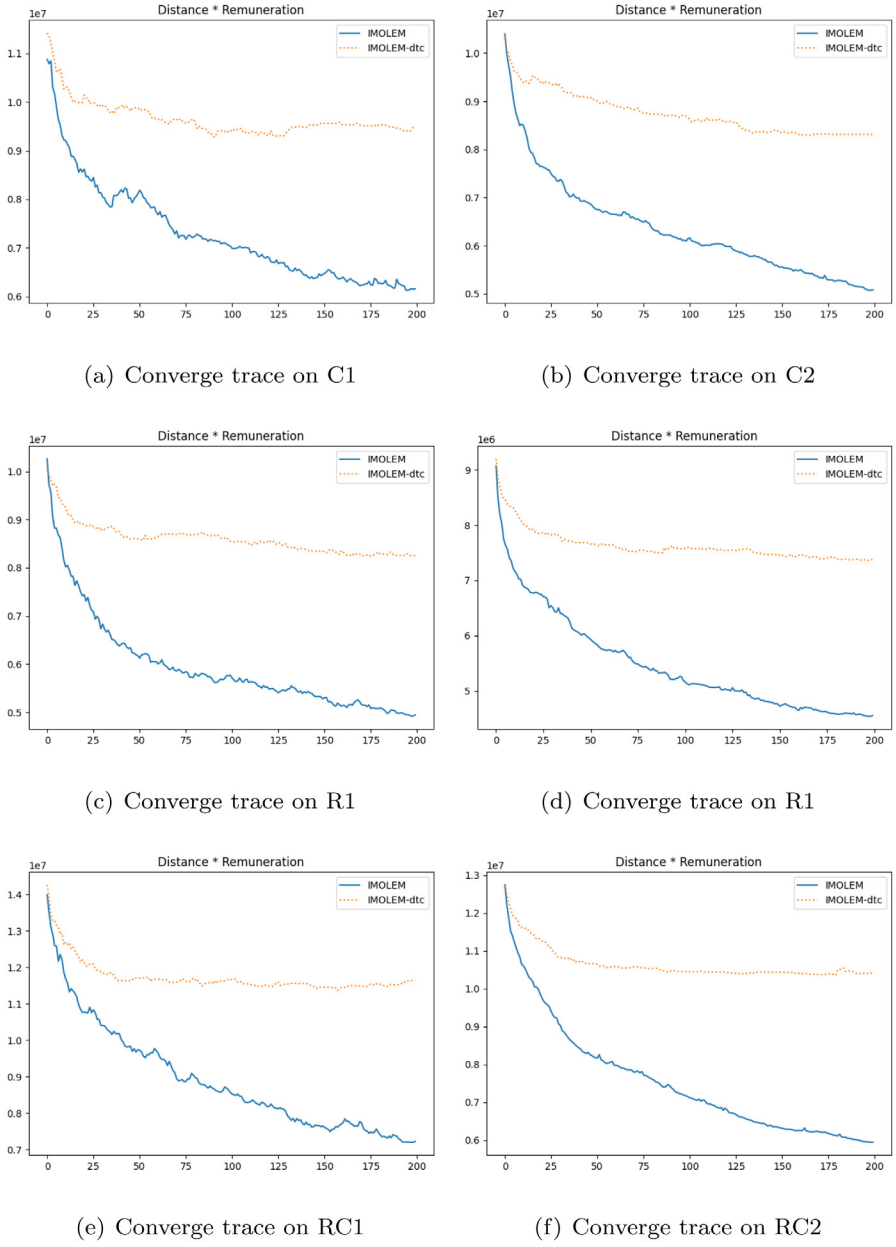
Table 3
Performance of IMOLEM and its three variants.

		NV	TD	DR	HV	Spacing
C1	IMOLEM	8.20	3152.07	1960.47	0.05	37.62
	Variant-1	7.72	3239.52	2068.00	0.08	38.95
	Variant-2	12.39	4008.50	2274.88	0.07	140.88
	Variant-3	15.95	4322.41	2413.42	0.07	57.18
C2	IMOLEM	8.95	2880.87	1734.37	0.08	36.64
	Variant-1	11.48	3016.17	1666.56	0.08	40.48
	Variant-2	14.38	3958.72	2143.22	0.05	69.76
	Variant-3	10.86	4162.36	2385.81	0.03	132.95
R1	IMOLEM	8.20	2535.08	1949.00	0.07	27.03
	Variant-1	9.79	2770.21	2023.52	0.06	33.80
	Variant-2	12.76	3428.15	2396.50	0.07	125.05
	Variant-3	15.64	3670.03	2374.67	0.07	160.33
R2	IMOLEM	11.00	2545.12	1746.85	0.05	36.43
	Variant-1	11.96	2565.86	1701.61	0.07	30.40
	Variant-2	10.43	3287.09	2425.75	0.04	71.77
	Variant-3	12.54	3512.59	2415.11	0.05	97.28
RC1	IMOLEM	10.00	3527.05	2041.93	0.06	34.48
	Variant-1	8.63	3525.49	2130.26	0.04	29.70
	Variant-2	13.17	4609.12	2580.22	0.05	75.17
	Variant-3	20.47	4856.41	2463.42	0.09	148.12
RC2	IMOLEM	10.04	3198.93	1827.90	0.08	37.65
	Variant-1	10.23	3206.84	1825.92	0.06	39.21
	Variant-2	12.64	4357.47	2467.71	0.04	75.95
	Variant-3	15.78	4649.83	2343.47	0.04	230.43

3.2.2. Hypothesis instantiating

The rules in H-group produce the priority value ranges for each customer. And new chromosomes created by following the rules are more likely to engender low-cost solution. Visiting the customers with higher priority values earlier than others may enhance the chance to yield a high quality solution to the MO-VRPSD. In contrast, visiting customers with lower priority values earlier than others may result in high-cost solution.

The decision tree classifier delivers both positive and negative rules. The positive rules will generate half evolutionary population size of chromosomes uniformly. For example, if there are 10 positive rules and evolutionary population size is 110, then each rule will instantiate 5 chromosomes, and the remaining 5 chromosomes will be generated by 5 randomly picked rules. Instantiating a chromosome is to satisfy the rules at the nodes of the decision tree. Each node has a priority range of the customer and the actual priority value is randomly selected within the feasible range. After all priority values of the customers on the nodes are allocated, the remaining customers will be also set to the priority values from the feasible range. However, in a chromosome, the priority values of customers should not be the same.



(a) Converge trace on C1 (b) Converge trace on C2

(c) Converge trace on R1 (d) Converge trace on R1

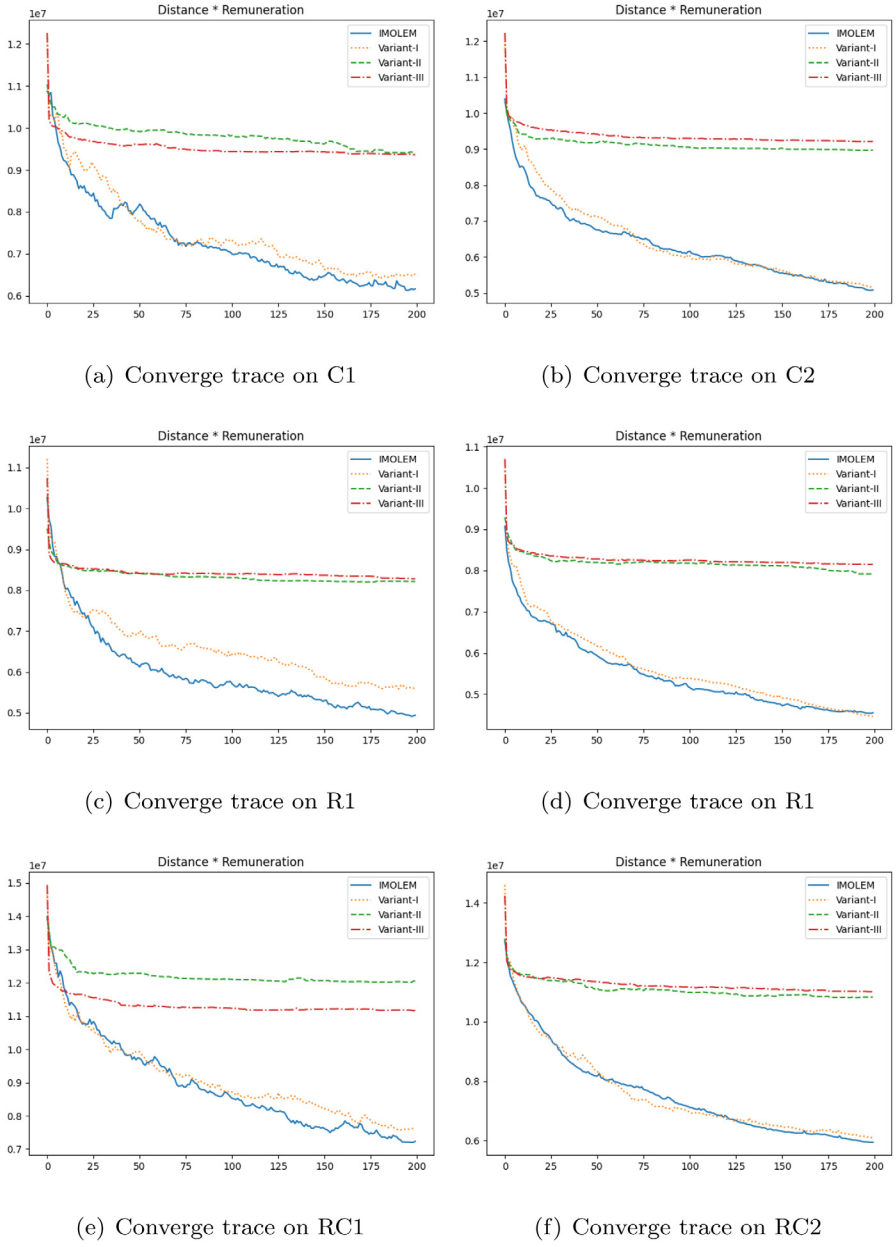
(e) Converge trace on RC1 (f) Converge trace on RC2

Fig. 5. Comparison of converge trace between IMOLEM and IMOLEM$_{_dtc}$.

After the instantiating process, some heuristic operations are applied to the individuals in the current first half of archive population before appending them to the archive set. The heuristic operations include *partial swapping*, *merging the shortest route*, *splitting the longest route* and *random shuffling*, which are associated with four parameters, i.e., search rate, swapping rate, merging rate, and shuffling rate. Concretely, partial swapping operation is to choose two sub-route in an individual, and a continuous segment randomly selected from each will swap with each other. Merging the shortest route operation is to choose two sub-routes in the solution with the lowest travel distance, and join them into one route. Splitting the longest route operation is to search and split the sub-route with the longest travel distance into two sub-routes at a random position. Random shuffle operation is to rearrange the sub-route in a random order. The four respective parameters govern the probability of occurrence for each operation. Fig. 4 shows the basic flow of the heuristic search operation. All chromosomes in the first half of archive set will apply the heuristic search operations regulated by the four parameters. These operations will lead to relatively large search space in the objective domain and may potentially prevent the solution search from being stuck in local optimums.

3.3. Route simulation method

The major difference between MO-VRPSD and its deterministic variant is that customers demands in the former are unknown until the vehicle actually arrives at the customer location. So is the occurrence of a route failure, and the actual cost will be revealed only when the route is realized. However, we assume that the demand distribution of the customer is available in advance, rely on which we could evaluate the approximate objective values before the actual realization of the route. And the average results of multiple repeated simulation could be used to evaluate the method. Particularly, the route simulation process is implemented as follows. We first generate N demands



(a) Converge trace on C1 (b) Converge trace on C2

(c) Converge trace on R1 (d) Converge trace on R1

(e) Converge trace on RC1 (f) Converge trace on RC2

Fig. 6. Comparison of converge trace among IMOLEM and its variants.

for each customer according to the respective distributions, and then repeat N times of route implementation and calculate its objective values in each run. Due to the stochasticity, the route failure may occur in a run while not in others. When the route failure occurs, it will incur additional travel distance and driver remuneration due to the replenishing policy. Accordingly, the objective value of a route are calculated as the average objective values of the N simulations.

After the heuristic instantiating process, a new population is generated and RSM is applied to each chromosome of the new population to attain their objective values. Then, the population is appended to the archive set, which is further sorted by the multi-objective optimization algorithm, i.e., ENS-SS. Finally, the archive set is truncated to the predetermined size. This process will repeat until the termination condition is satisfied.

4. Experiments and results

We conduct experiments to verify our method, i.e., IMOLEM, by evaluating the respective components and the overall framework. Given the stochastic characteristic, there is no common benchmark to test the MO-VRPSD problem. So we select six representative instances of standard VRPTW from Solomon benchmark [37], i.e., C1, C2, R1, R2, RC1 and RC2, based on which we generate the data required for MO-VRPSD. Specifically, the locations of depot and customers are the same to the ones in Solomon benchmark. The mean demand of each customer is set to the respective deterministic one in Solomon benchmark, and the standard deviation is set to a random portion of the mean demand, which is uniformly sampled from [0,1/3]. And the remuneration of driver is set to $10 per normal hour and $20 per overtime hour. The normal working hour is calculated according to the specific map used. The time window B is set to 0.8 times the depot "due date" in Solomon benchmark. This

Table 4
Performance comparison for different optimization criteria.

		Scale	NV	TD	DR	HV	Spacing
C1	MO	25	8.20	3152.07	1960.47	0.05	37.62
	DTDDR	12	16.42	3108.98	1390.69	0.01	13.20
	DTDNV	18	6.94	3032.66	1985.71	0.04	39.76
	DDRNV	33	9.12	3161.60	1905.16	0.07	29.50
	STD	8	13.50	2777.39	1353.45	0.01	18.28
	SDR	2	18.50	3132.86	1345.26	0.00	0.00
	SNV	1	1.00	3951.63	3048.11	0.00	0.00
C2	MO	22	8.95	2880.87	1734.37	0.08	36.64
	DTDDR	12	13.58	3074.42	1510.92	0.01	21.23
	DTDNV	9	2.67	2768.65	2094.75	0.01	32.51
	DDRNV	17	8.29	2889.58	1755.24	0.05	40.51
	STD	8	4.00	2704.04	1961.50	0.02	34.44
	SDR	2	17.50	3062.80	1314.39	0.00	0.00
	SNV	1	1.00	3447.72	2667.27	0.00	0.00
R1	MO	25	8.20	2535.08	1949.00	0.07	27.03
	DTDDR	12	14.33	2677.55	1589.99	0.01	32.83
	DTDNV	9	3.44	2547.89	2303.53	0.02	30.83
	DDRNV	29	10.14	2701.37	1933.30	0.05	33.74
	STD	9	9.89	2457.78	1720.59	0.01	30.73
	SDR	2	19.50	2743.39	1429.93	0.00	0.00
	SNV	1	1.00	3265.87	3070.39	0.00	0.00
R2	MO	23	11.00	2545.12	1746.85	0.05	36.43
	DTDDR	19	9.89	2483.17	1765.24	0.05	34.87
	DTDNV	4	2.50	2198.02	2050.72	0.01	21.84
	DDRNV	14	8.29	2458.19	1838.75	0.07	46.62
	STD	4	2.50	2091.48	1960.30	0.01	42.82
	SDR	3	20.67	2780.78	1438.76	0.00	15.20
	SNV	1	1.00	2970.26	2764.35	0.00	0.00
RC1	MO	29	10.00	3527.05	2041.93	0.06	34.48
	DTDDR	13	16.62	3556.57	1614.82	0.01	63.04
	DTDNV	11	4.18	3296.69	2351.64	0.02	58.38
	DDRNV	29	9.38	3510.80	2088.78	0.08	31.99
	STD	9	11.78	3274.60	1729.42	0.02	79.36
	SDR	4	20.50	3479.63	1449.77	0.00	20.64
	SNV	2	1.00	4430.70	3342.25	0.00	0.00
RC2	MO	25	10.04	3198.93	1827.90	0.08	37.65
	DTDDR	19	10.32	3238.55	1823.87	0.04	25.70
	DTDNV	5	3.00	2762.97	2037.86	0.01	10.78
	DDRNV	24	13.12	3446.88	1784.99	0.08	36.97
	STD	6	2.67	2731.51	2043.07	0.01	45.42
	SDR	1	21.00	3588.08	1477.58	0.00	0.00
	SNV	1	1.00	3967.72	2968.25	0.00	0.00

Table 5
Comparison of the IMOLEM and MOEA.

		NV	TD	DR	HV	Spacing
C1	IMOLEM	8.20	3152.07	1960.47	0.05	37.62
	MOEA	11.23	3259.00	1904.34	0.09	97.20
	MRDL	17.18	3238.56	1526.51	0.05	468.11
C2	IMOLEM	8.95	2880.87	1734.37	0.08	36.64
	MOEA	12.00	2969.60	1724.60	0.11	131.93
	MRDL	10.82	3129.12	1807.84	0.10	328.57
R1	IMOLEM	8.20	2535.08	1949.00	0.07	27.03
	MOEA	9.19	3065.71	2321.43	0.10	124.62
	MRDL	18.56	2851.21	1598.30	0.03	492.42
R2	IMOLEM	11.00	2545.12	1746.85	0.05	36.43
	MOEA	10.50	2818.46	2049.02	0.09	49.46
	MRDL	13.50	2819.40	1844.03	0.06	241.71
RC1	IMOLEM	10.00	3527.05	2041.93	0.06	34.48
	MOEA	12.85	3546.56	1966.13	0.12	97.45
	MRDL	13.20	3906.95	2088.18	0.16	704.68
RC2	IMOLEM	10.04	3198.93	1827.90	0.08	37.65
	MOEA	11.45	3593.72	2055.10	0.11	42.87
	MRDL	14.36	3644.75	1875.48	0.05	710.24

time window refers to the normal working duration, and it is equivalent to eight hours in reality, which is used to compute the driver remuneration according to Eq. (4). Our method was implemented in Python 3 on a machine with Intel Core i5-8600K. Table 1 shows the parameter settings in the experiments.

4.1. Effectiveness of learning component

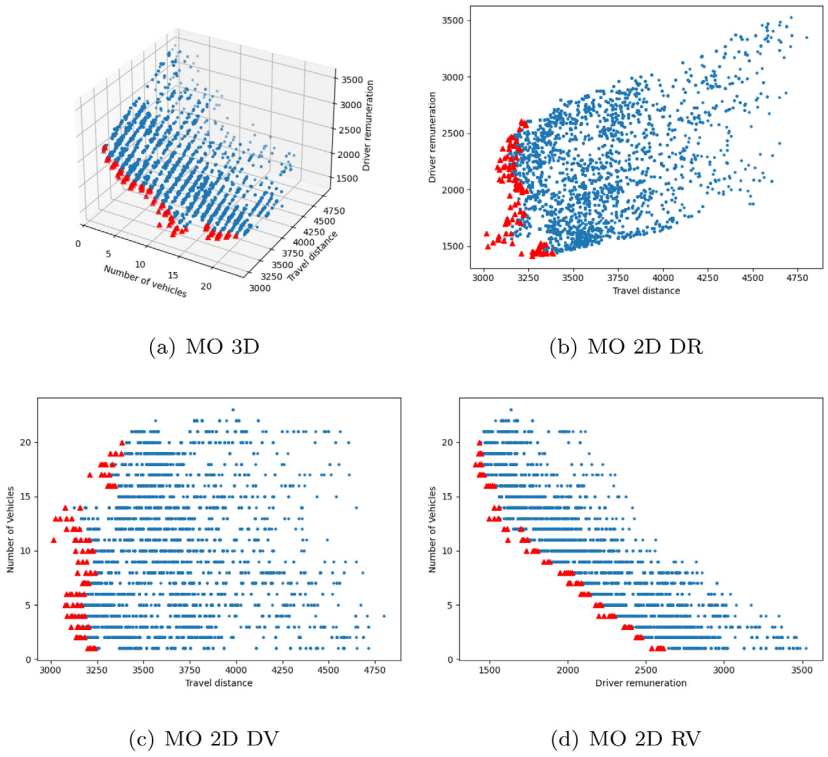
One major difference between our IMOLEM and traditional evolutionary algorithm is that the former incorporates a machine learning algorithm, i.e., decision tree, to guide the generation of new populations. To verify the effectiveness of the decision tree in the instantiating process, we compare our IMOLEM with IMOLEM$_{-dtc}$, the latter of which performs the same initialization process while disabling the decision tree classifier in the instantiating process. To compare the quality of different solution sets when their objective values are close, two more quality metrics for multi-objective are introduced, i.e., hypervolume (HV) [38] and spacing [39]. In specific, HV is an indicator to evaluate the multi-objective solution set, and it measures the volume covered by solutions of a non-dominated set in the objective space. In general, the larger the HV, the better the solution set. The spacing is an indicator to evaluate the distributivity of a population, and a smaller spacing value usually implies a well-distributed population. Then, IMOLEM and IMOLEM$_{-dtc}$ are tested on the six selected instances and adopt the identical parameter settings as in Table 1 with the same number of generations.

The experimental results are displayed in Table 2, which include the average of three objective values, i.e., travel distance (TD), driver remuneration (DR) and number of vehicles (NV). Moreover, the HV and spacing of the non-dominated solution sets are also recorded. From Table 2, we can observe that IMOLEM$_{-dtc}$ will converge into a solution set with lower number of vehicles but much higher travel distance and driver remuneration. The spacing metrics of IMOLEM$_{-dtc}$ are all inferior to that of IMOLEM. Fig. 5 depicts the curves of TD and DR for both IMOLEM and IMOLEM$_{-dtc}$ on the six instances. The comparisons were performed based on the product of average travel distance and average driver remuneration of all solutions [40]. The convergence curve is to record the population in each iteration and calculate the average objective values of all solutions in the population. From Fig. 5, it is obvious to see that IMOLEM has a higher convergence speed on each instance and the solution sets of IMOLEM are also superior to IMOLEM$_{-dtc}$. The solution sets of IMOLEM$_{-dtc}$ converged to a higher product of TD and DR while the NV is close. In general, applying the decision tree classifier in the evolutionary process enables a higher convergence speed with solution set of better quality.

4.2. Effectiveness of heuristic operators

In our IMOLEM, we employ some heuristic operators to attain a better solution set in the initialization and instantiating process. We conduct experiments to verify the effectiveness of the respective heuristic operators. To this end, we also present three variants of IMOLEM and compare them with the original one. Specifically, *Variant-1* is dismounted of the heuristic initialization operator, *Variant-2* is dismounted of the heuristic instantiating operator, and *Variant-3* is dismounted of both the two operators. In Variant-1, the initial population is generated with complete randomness. In Variant-2, the population is generated based on the hypothesis but does not apply the heuristic instantiating operator. Variant-3 is a combination of variant-1 and variant-2. Other than that, IMOLEM and all the variants adopt the same parameter settings in Table 1, and run on the same six instances independently.

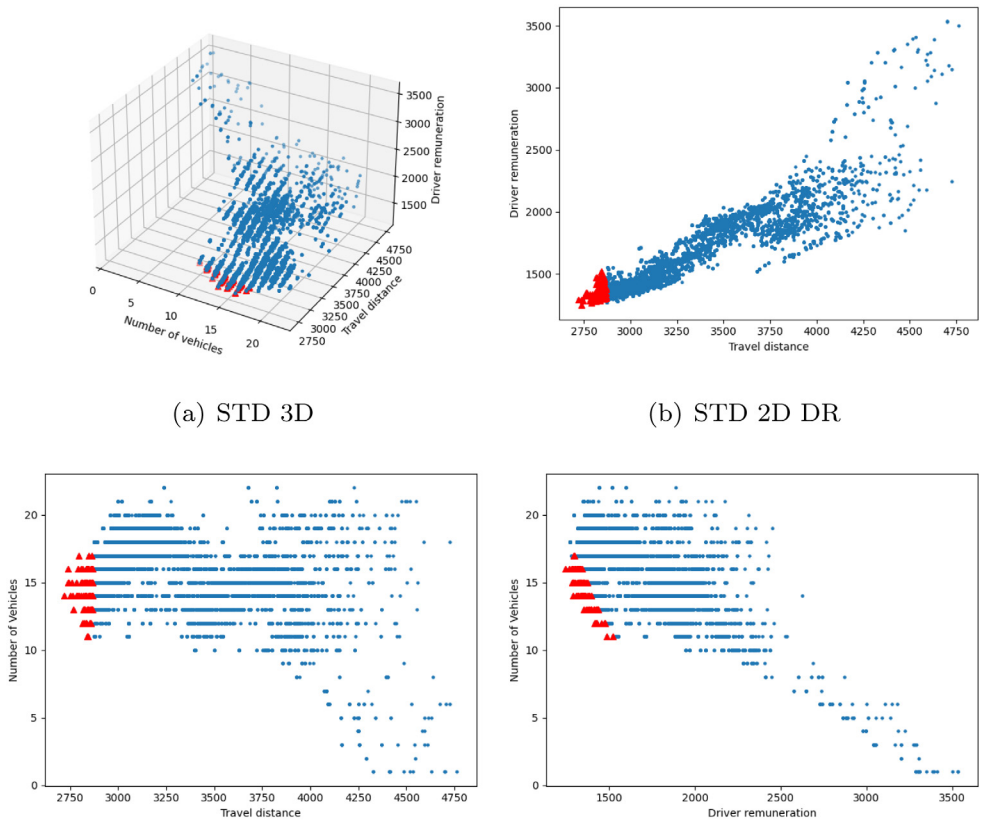
The experimental results are recorded in Table 3, which include the three average objective values. Besides, the HV and spacing of the non-dominated solution sets are also included.



(a) MO 3D (b) MO 2D DR

(c) MO 2D DV (d) MO 2D RV

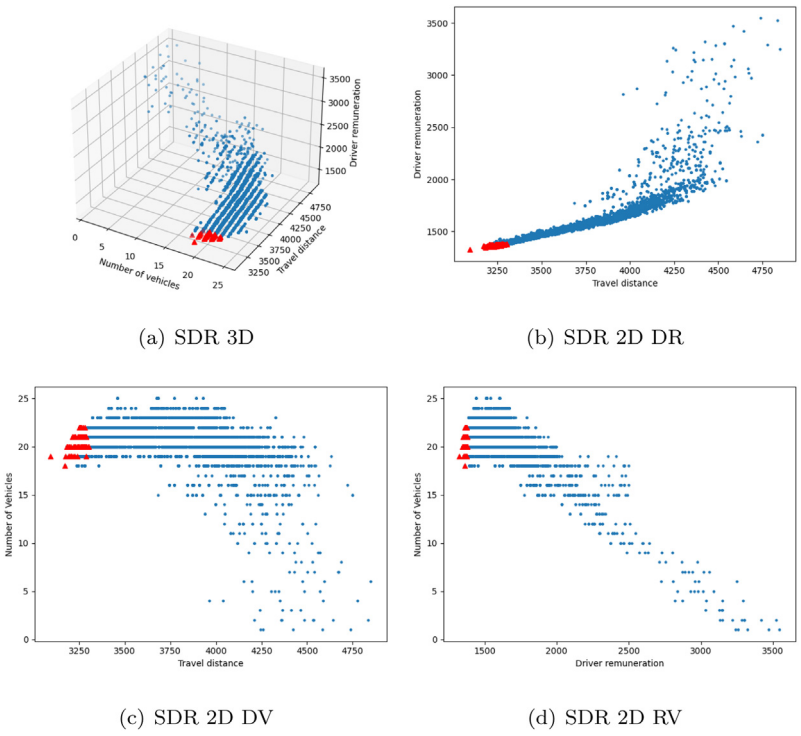
Fig. 7. Search space of MO on C1.



(a) STD 3D (b) STD 2D DR

(c) STD 2D DV (d) STD 2D RV

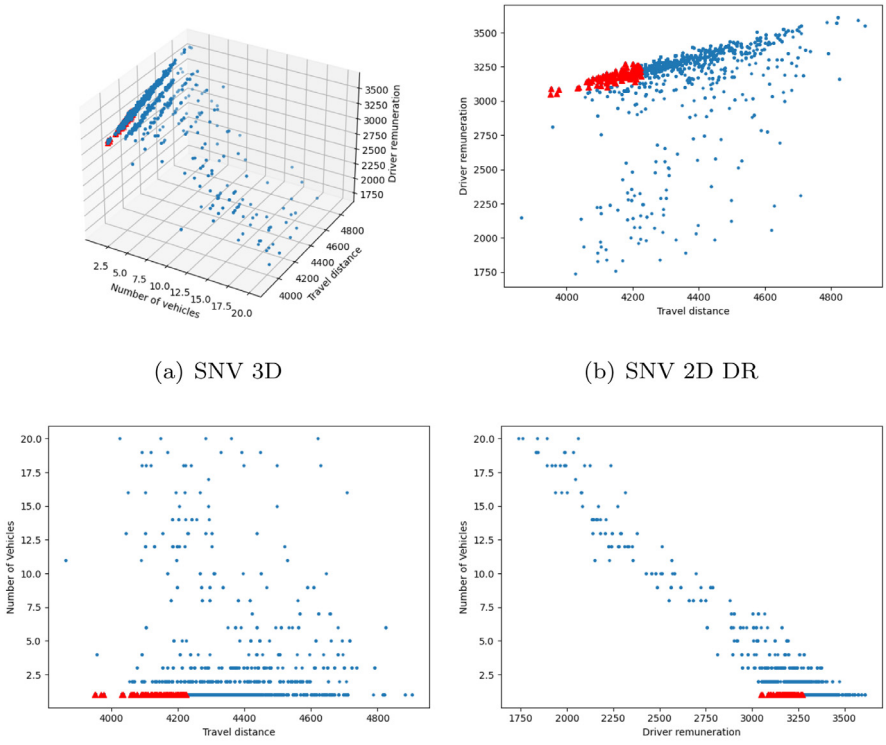
Fig. 8. Search space of STD on C1.



(a) SDR 3D (b) SDR 2D DR

(c) SDR 2D DV (d) SDR 2D RV

Fig. 9. Search space of SDR on C1.

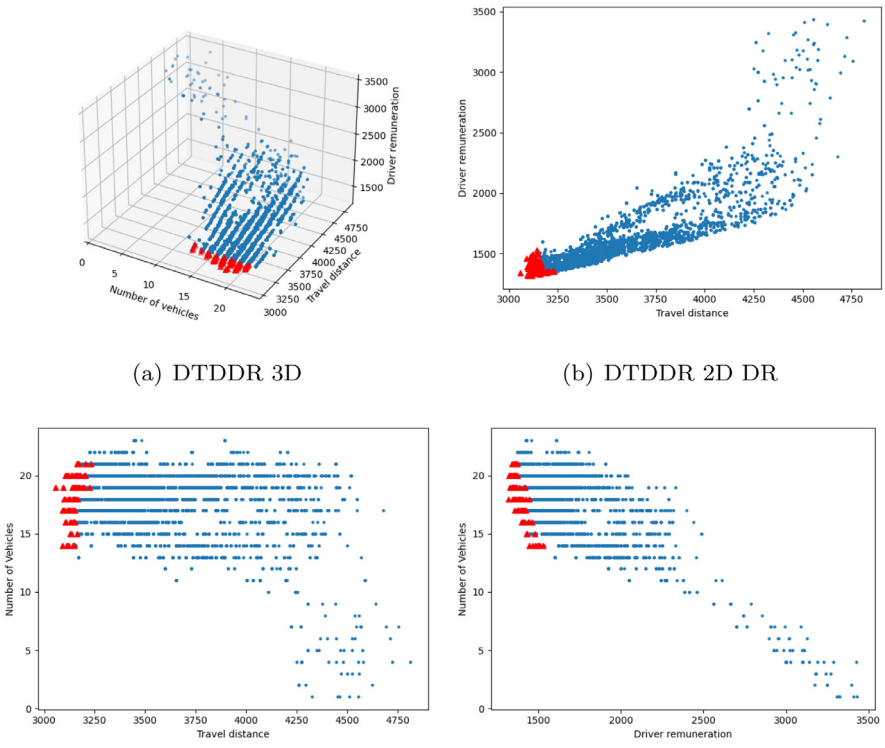


(a) SNV 3D (b) SNV 2D DR

(c) SNV 2D DV (d) SNV 2D RV

Fig. 10. Search space of SNV on C1.

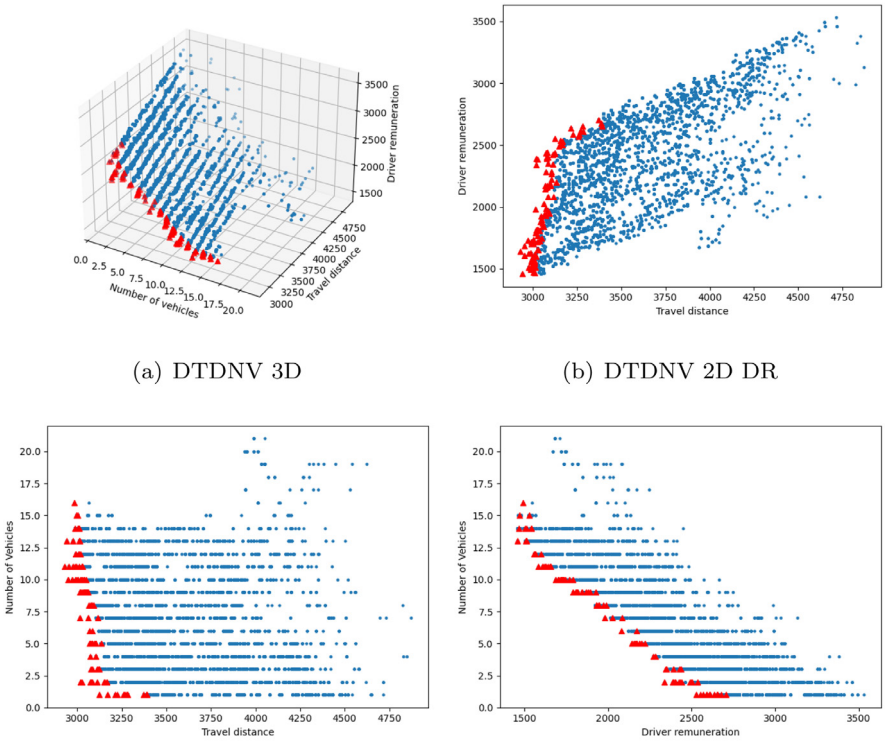
From Table 3, we observe that the quality of solutions obtained by our proposed IMOLEM is higher than that of three variants. Notably, the average objective values of the three variants for the six instances are mostly dominated by the IMOLEM. Variant-2 tends to find a solution set with relative smaller NV but its TD and DR are much higher than IMOLEM, and its spacing indicators are large which means its population distribution is terrible. So the three variants are unable to converge to a suitable population in less iterations.



(a) DTDDR 3D (b) DTDDR 2D DR

(c) DTDDR 2D DV (d) DTDDR 2D RV

Fig. 11. Search space of DTDDR on C1.

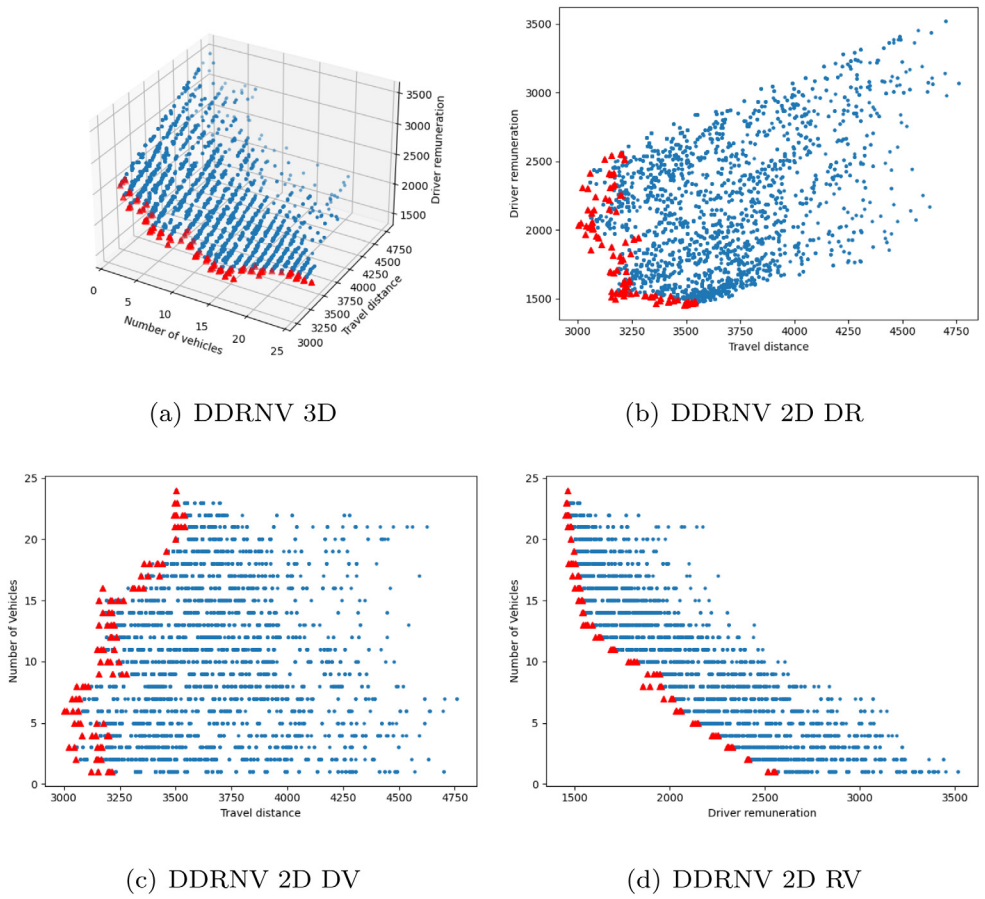


(a) DTDNV 3D (b) DTDNV 2D DR

(c) DTDNV 2D DV (d) DTDNV 2D RV

Fig. 12. Search space of DTDNV on C1.

Fig. 6 presents the convergence curves of TD and DR for IMOLEM and the three variants on the six instances. And the multiplicative aggregation of average travel distance and average driver remuneration of all solutions is adopted. It is salient that



(a) DDRNV 3D (b) DDRNV 2D DR

(c) DDRNV 2D DV (d) DDRNV 2D RV

Fig. 13. Search space of DDRNV on C1.

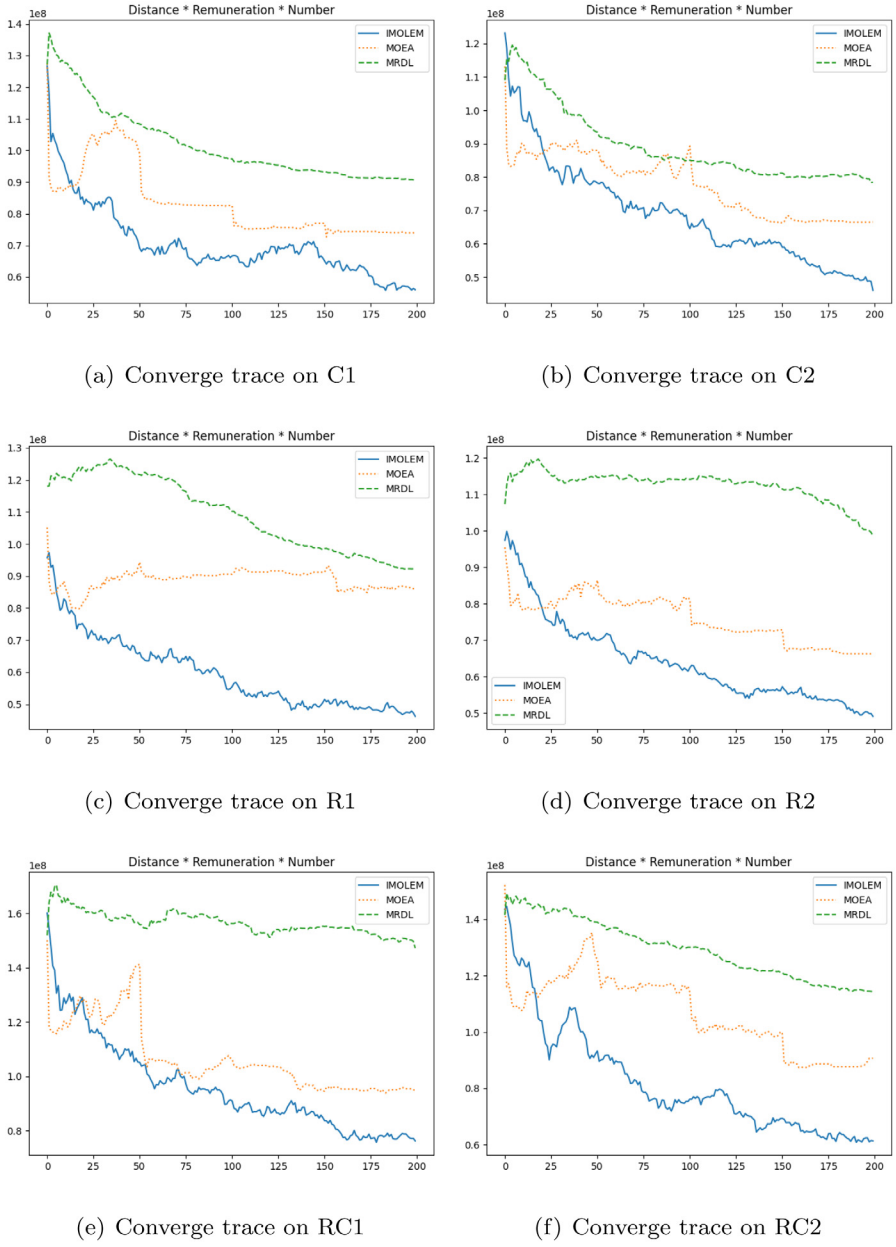
IMOLEM has a higher converging speed and better initial population. The result of IMOLEM is also superior compared with the three variants. Variant-1 has no heuristic initial population generator, so its initial population performs inferior to others and may consume higher computation cost for convergence. Variant-2 has no heuristic instantiating operator, which demonstrates lower converging speed and may also run into the population of local optimum. Variant-3 does not have either of the heuristic operators and performs the most inferior as expected.

4.3. The performance of multi-objective optimization algorithm

In our IMOLEM, we leverage a Pareto sort approach to better explore the objective space. There are three objectives in our MO-VRPSD to be considered simultaneously. In reality, we might not only be interested in one of those objectives, but also in feasible solutions with relatively small travel distance, driver remuneration and number of vehicles, so that we could choose one of the solutions according to the preference in a given actual situation. In the literature, the VRPSD is usually solved by optimizing one particular objective respectively or exploiting a linear combination of the objectives and converting it to a single-objective problem. However, the drawbacks of such methods are inevitable as the weights of the linear combination are hard to be determined, which is inappropriate to the real-world complex vehicle routing problems. Obviously, this issue can be considerably mitigated if the three objectives are optimized concurrently without the need of determining their weightages. To verify the effectiveness of the multi-objective optimization algorithm in our IMOLEM, we investigate six more different settings for coping with the respective optimization objectives. Specifically, three of them are single objectives, which are minimizing travel distance (STD), driver remuneration (SDR), and number of vehicles (SNV), respectively. The other three of them are minimizing two objectives simultaneously, i.e., travel distance and driver remuneration (DTDDR), travel distance and number of vehicles (DTDNV), and driver remuneration and number of vehicles (DDRNV), respectively. Besides, optimizing all of the three objectives simultaneously (MO) refers to our method.

Table 4 records the solution set results of different optimization settings on the six instances. Besides the five previously adopted metrics, we also consider the *scale*, which denotes the number of non-dominated solutions selected from the last population. From Table 4 it can be observed that, MO can produce the best non-dominated solution set with the largest scale value which serves as the foundation of a well-distributed diversity. Although STD, SDR and SNV can yield a better respective single objective, they leave other objectives considerably unsatisfactory. The solution set they produced might be infeasible given the inferior objective values and small scale. Especially, SNV on five instances only produces one single solution and two solutions on the remaining instance. The bi-objective types have similar weaknesses, although their solution set scales are larger than the single objective type. Nevertheless, they are still much smaller in comparison with the MO. The bi-objective types focus on their own two objectives and neglect the remaining one, which may lead to an undesirable evolutionary direction. Thus, they still may not perform comprehensive search in the objective domain space.

The search spaces of these different optimization types are depicted in Figs. 7–13. Each point in the 3-dim plots refers to a solution which has been explored by the respective type during the evolutionary process. Red triangle points denote the last archive



(a) Converge trace on C1 (b) Converge trace on C2

(c) Converge trace on R1 (d) Converge trace on R2

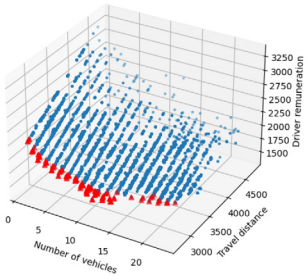
(e) Converge trace on RC1 (f) Converge trace on RC2

Fig. 14. Comparison of converge trace between IMOLEM and MOEA on C1.

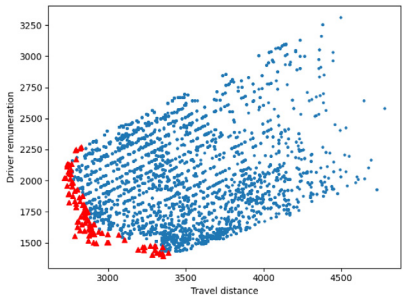
population. To demonstrate the relationship between any two objectives, additional separate 2-dim plots are also presented. Note that each 3-dim plot relates to three corresponding 2-dim plots. From these plots we can see that, the objective space searched by MO is considerably larger than the other four optimization types except DTDNV and DDRNV. These four optimization types have obvious vacancy in the objective domain, and they cannot perform a comprehensive search and may get trapped in local optima, which also explains why they cannot achieve a feasible solution set. From Figs. 7, 12 and 13, it can be observed that the three optimization types have similar search space. Both DTDNV and DDRNV having a competitive search space compared to MO suggests that driver remuneration and travel distance are not thoroughly conflicted objectives. A long travel distance usually leads to a high driver remuneration, and they are positively correlated but have different growth rates. Therefore, it might be possible to minimize both of them concurrently. In contrast, driver remuneration and travel distance are conflicted with the number of vehicles, and the attempt to minimize number of vehicles may lead to an increase in the other two objectives.

4.4. Comparison with other algorithms

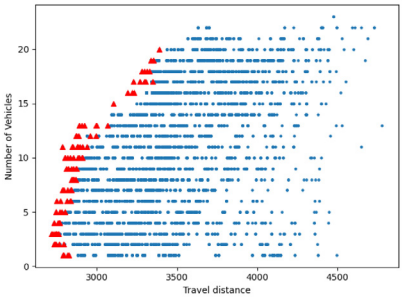
To verify the effectiveness of the overall algorithm of our IMOLEM, we compare it with another two strong algorithms for solving the MO-VRPSD, i.e., MOEA [32] and MRDL [33]. To this end, we run the three methods on the same six modified Solomon instances with identical generations. Regarding the those instances, customers in C1 are clustered around the central depot and they have short time windows. Customers in C2 have the same coordinates as C1 but have long time windows. The customers in R1 are distributed randomly and have short time windows, while long time windows in R2. RC1 is a combination of R1 and C1, where customers are distributed both randomly and clustered with short time windows. And the customers in RC2 have long time windows. Although some modifications are made



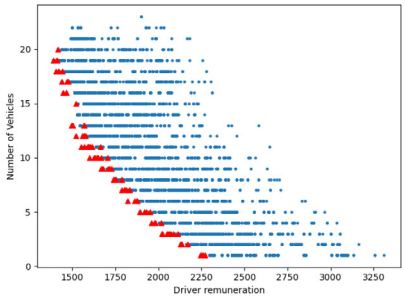
(a) IMOLEM search space 3D



(b) IMOLEM search space 2D DR

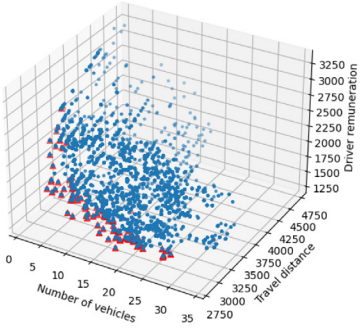


(c) IMOLEM search space 2D DV

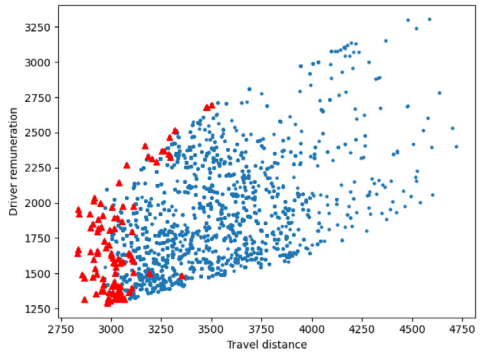


(d) IMOLEM search space 2D RV

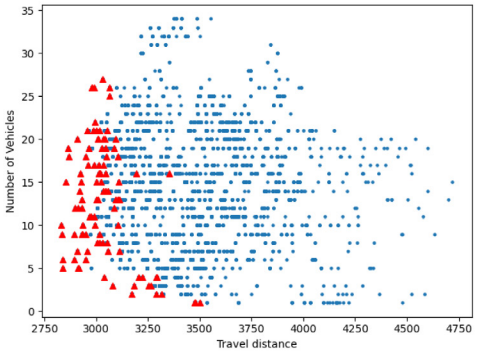
Fig. 15. Search space of IMOLEM on C2.



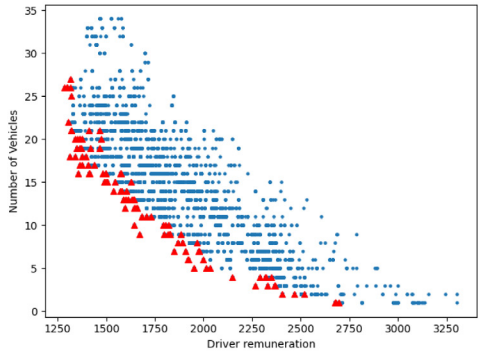
(a) MOEA search space 3D



(b) MOEA search space 2D DR

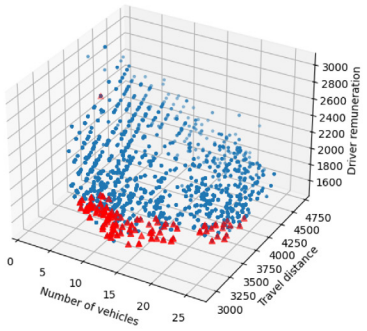


(c) MOEA search space 2D DV

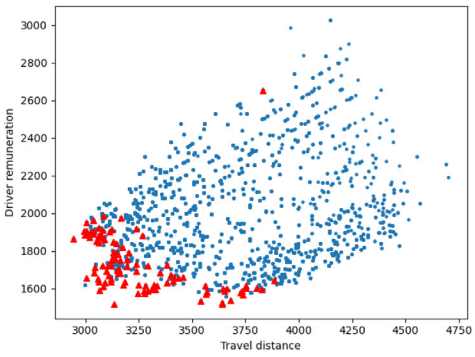


(d) MOEA search space 2D RV

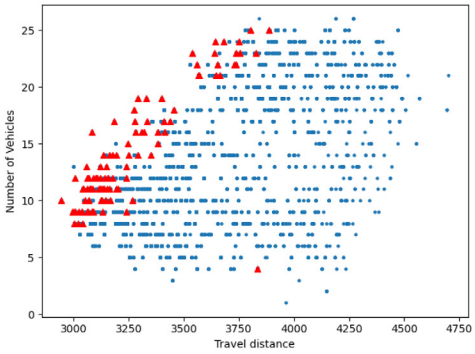
Fig. 16. Search space of MOEA on C2.



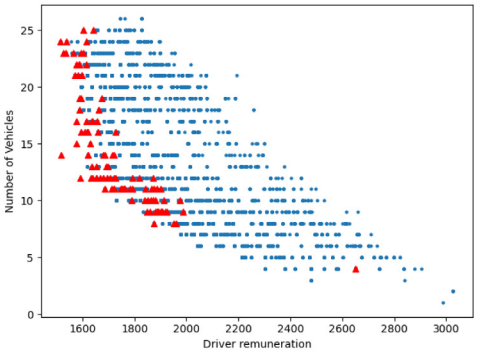
(a) MRDL search space 3D



(b) MRDL search space 2D DR

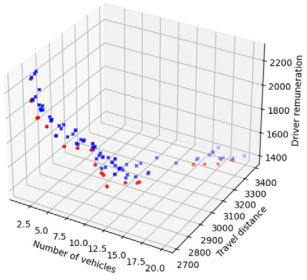


(c) MRDL search space 2D DV

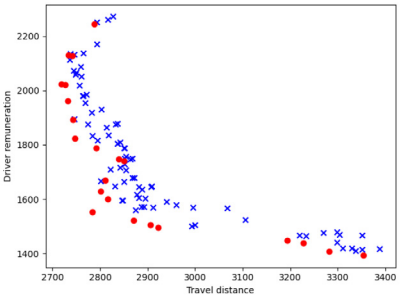


(d) MRDL search space 2D RV

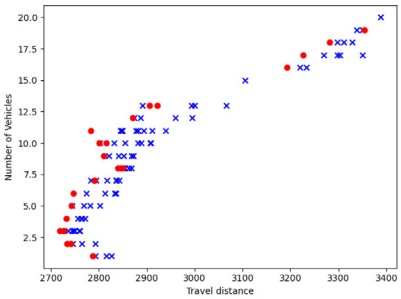
Fig. 17. Search space of DBMOEA on C2.



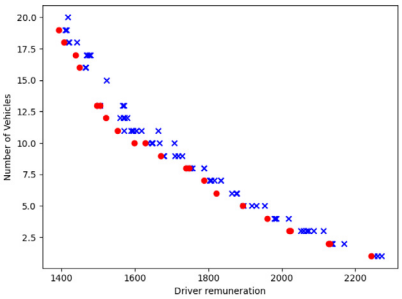
(a) LEM archive population 3D



(b) LEM archive population 2D DR

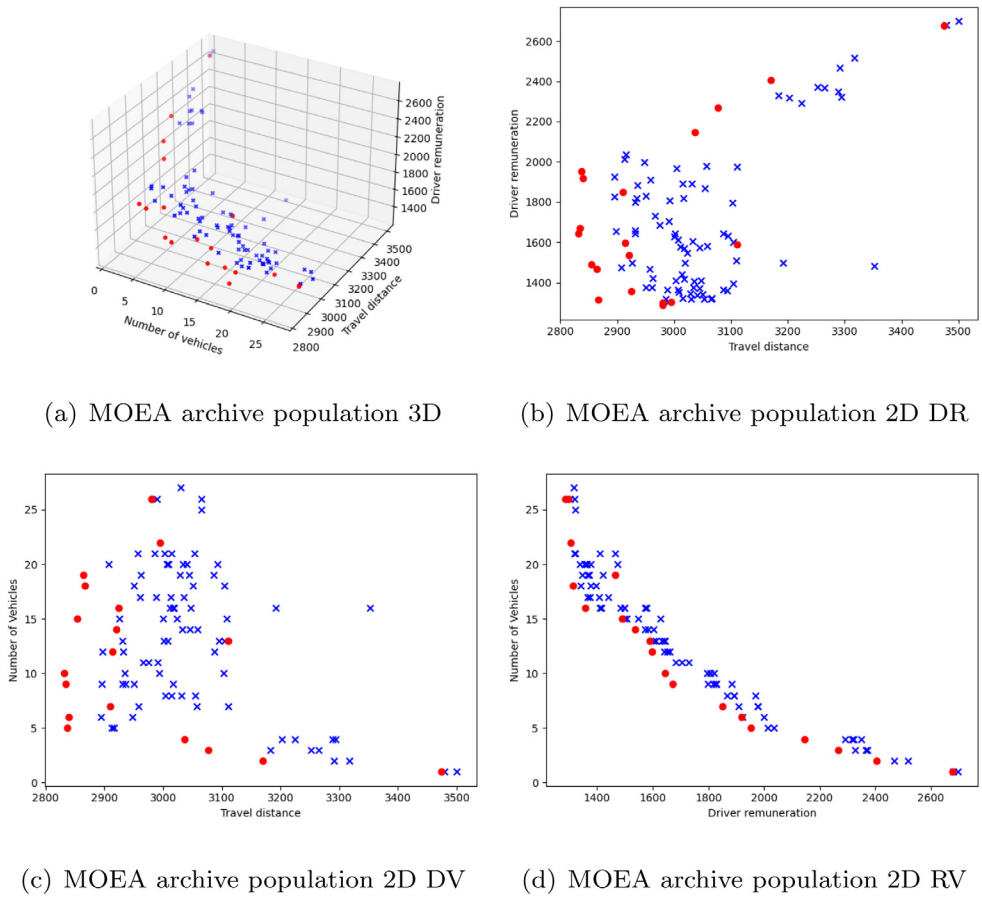


(c) LEM archive population 2D DV



(d) LEM archive population 2D RV

Fig. 18. Final archive population of IMOLEM on C2.



(a) MOEA archive population 3D (b) MOEA archive population 2D DR

(c) MOEA archive population 2D DV (d) MOEA archive population 2D RV

Fig. 19. Final archive population of MOEA on C2.

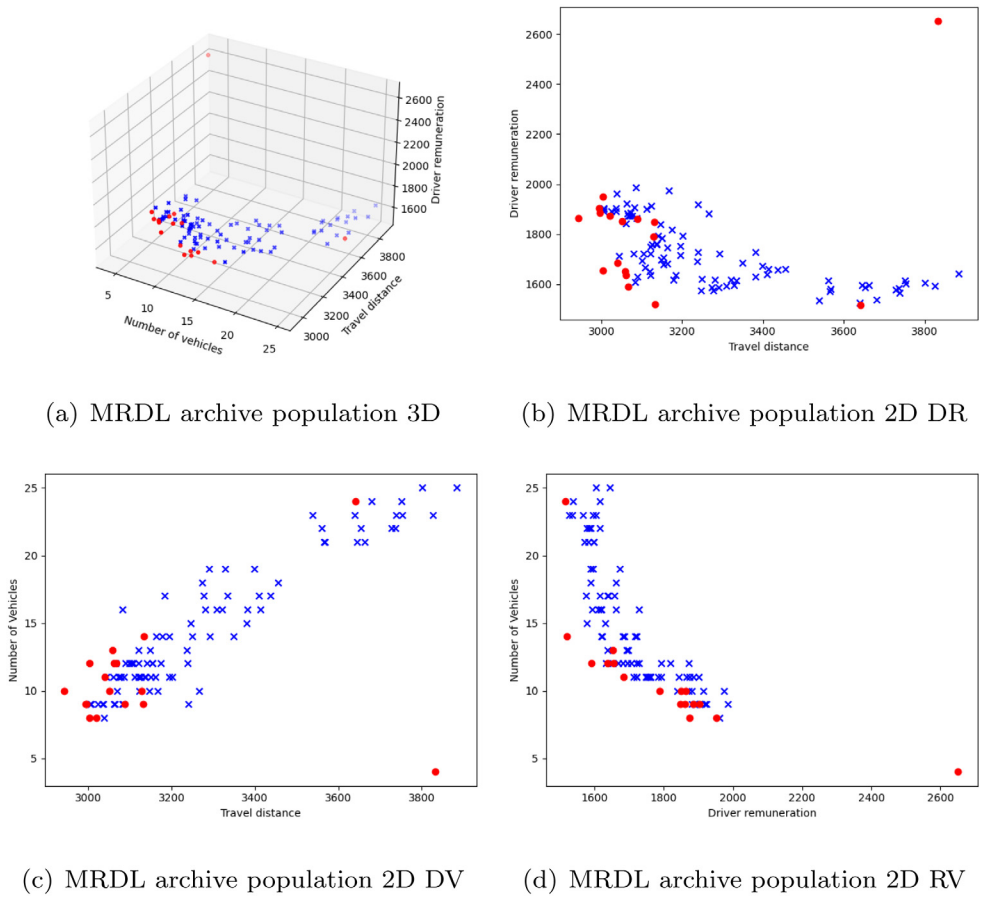
to fit MO-VRPSD, they still have the same geographical distribution characteristic. In short, C1 and C2 are central depot type, R1 and R2 are randomly distribution type, and RC1 and RC2 are both random and clustered distribution type. The experimental results are recorded in Table 5.

From Table 5, it can be seen that the objective values of IMOLEM are superior in most situations. On R1 and RC2, the solutions of IMOLEM dominate MOEA. In other situations, the solutions do not dominate, as IMOLEM prefers to sacrifice a bit TD and DR to attain better NV. On every instance, IMOLEM demonstrates better spacing, which means that IMOLEM is able to produce a solution set with satisfactory distribution. MOEA performs better on HV with IMOLEM. Nevertheless, IMOLEM has better performance on the average objective values of non-dominated solution set, which nearly dominates the solution of MOEA and MRDL. The objective values of MRDL seems bad, but on C1 and R1, it obtains much more better DR objective. MRDL is based on decomposition strategy, it is not suitable to deal with objectives where the magnitude of the objective value is too large. In general, IMOLEM is able to yield a superior non-dominated solution set in comparison with MOEA and MRDL.

Fig. 14 presents the convergence curves of IMOLEM, MOEA and MRDL. The product of three objective values: average travel distance, average driver remuneration and average number of vehicles of each archive generation is plotted on the same six problem instances. It is notable that IMOLEM has higher convergence speed than MOEA and MRDL. Specifically, IMOLEM converges considerably faster on C1, C2, R1, R2 and RC1. And on RC2, IMOLEM still converges faster, but not as salient as on other instances. MRDL is not suitable for optimizing number of vehicles target, it can be better optimized for the other two objectives, especially driver remuneration, but the number of vehicles cannot be fully optimized, so the product with number of vehicles is always worse than other two algorithms. Local search methods are adopted every 50 generations in MOEA, and it is obvious that MOEA has a saltation convergence every 50 generations. Superiority of IMOLEM on higher convergence speed may come from the decision tree, which extracts the characteristic of high performance solutions and low performance solutions, and leads the evolutionary process to a desirable direction.

Figs. 15–17 show the search space of three algorithms during the whole iteration process on C2 instance. In the diagram, each point represents a search result of three objective values during the evolution and red dots represent the last generation of the archive population. As can be seen from the figures, IMOLEM has a more uniform search space during the evolution, and the last generation is also spread out across the objective space. This can also be seen in Table 5, where IMOLEM has lower spacing indicator, which means it has better diversity in the population. The search spaces of MOEA and IMOLEM are similar, however, the search space of MOEA is less uniform than that of IMOEM, especially from the last generation of population, where the search direction of MOEA is more uneven. On the other hand, MRDL is more inclined to the target with larger value in the search process and has poor optimization effect for the vehicle number with lower value.

The final archive population of three algorithms on C2 instance are plotted in Figs. 18–20. In these figures, red dots represent the solutions in the final archive population that are not dominated by any other solutions and can be used as the final decision scheme, blue cross represents other individuals in the final archive population. As can be seen from the figures, IMOLEM



(a) MRDL archive population 3D (b) MRDL archive population 2D DR

(c) MRDL archive population 2D DV (d) MRDL archive population 2D RV

Fig. 20. Final archive population of MRDL on C2.

is close to a frontier between any objectives, which implies a desirable state of convergence. MOEA only shows a good convergence state between the two objectives while it is messy between other objectives. As for MRDL, it can be clearly seen that its optimization direction for the number of vehicles is missing, and the MRDL does not retain solutions in the space with a lower number of vehicles.

The experimental results justify that our IMOLEM is able to achieve superior performance for solving the MO-VRPSD. It is able to produce a high quality solution set in relatively short computational time. The reason for the performance boosting by IMOLEM comes from the advantage of the new chromosome representation scheme, which allows the LEM to guide the evolutionary direction by using the knowledge it learned during the evolution process, and find more promising direction towards the optimal solutions. In the meantime, the heuristic operations in the initialization and instantiating process are also important, as they potentially expand the search space so that the evolution process will effectively avoid to get trapped in local optima.

5. Conclusions

In this paper, we presented IMOLEM to solve the MO-VRPSD. The IMOLEM exploited a decision tree algorithm to effectively guide the direction of evolution by leveraging the knowledge learned during the evolutionary process. More importantly, to ensure unique map between chromosome representation and corresponding MO-VRPSD solutions, we proposed a novel priority-based encoding and decoding approach with bubbles. Besides, the ENS-SS was incorporated to sort the multi-objective individuals and generate the Pareto set, which also served as an efficient Pareto sorting function. Additionally, two heuristic operators adopted in the initialization and instantiating process enhanced the performance of the IMOLEM for generating initial population and instantiating the hypothesis. We conducted various experiments on modified Solomon benchmark to evaluate our proposed IMOLEM. The effectiveness of decision tree classifier and heuristic operators in the initialization and instantiating process were justified by the experimental results. The IMOLEM was also compared with other efficient evolutionary algorithms, and it showed that IMOLEM could deliver superior performance in terms of both solutions quality and computational time.

Regarding the future works, we will study the follows aspects. (1) To make the approach more handy, we will investigate adaptive approach to automatically configure the parameters. (2) We will modify and improve IMOLEM so that it is able to solve different types of vehicle routing problems. (3) We will test our method on more different benchmarks or real-world datasets. (4) We will also consider more intelligent algorithms based deep learning or deep reinforcement learning [41–43] to solve the multi-objective VRPSD.

CRediT authorship contribution statement

Yunyun Niu: Conceptualization, Methodology. **Detian Kong:** Software, Writing – original draft. **Rong Wen:** Methodology. **Zhiguang Cao:** Methodology, Writing – review & editing. **Jianhua Xiao:** Funding acquisition, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the National Natural Science Foundation of China [grant numbers 61872325, 62072258, 61772290, 61803104]; the Fundamental Research Funds for the Central Universities, China [grant number 2652019028]; and the China Scholarship Council [grant number 201806405004].

References

[1] G. Laporte, The vehicle routing problem: An overview of exact and approximate algorithms, European J. Oper. Res. 59 (3) (1992) 345–358.
[2] E. Berhan, B. Beshah, D. Kitaw, A. Abraham, Stochastic vehicle routing problem: a literature survey, J. Inform. Knowl. Manag. 13 (03) (2014) 1450022.
[3] W.-H. Yang, K. Mathur, R.H. Ballou, Stochastic vehicle routing problem with restocking, Transp. Sci. 34 (1) (2000) 99–112.
[4] G. Laporte, F.V. Louveaux, Solving stochastic routing problems with the integer L-shaped method, in: Fleet Management and Logistics, Springer, 1998, pp. 159–167.
[5] M. Gendreau, G. Laporte, R. Séguin, Stochastic vehicle routing, European J. Oper. Res. 88 (1) (1996) 3–12.
[6] C.Y. Cheong, K.C. Tan, D. Liu, J.-X. Xu, A multiobjective evolutionary algorithm for solving vehicle routing problem with stochastic demand, in: 2006 IEEE International Conference on Evolutionary Computation, IEEE, 2006, pp. 1370–1377.
[7] D.J. Bertsimas, P. Jaillet, A.R. Odoni, A priori optimization, Oper. Res. 38 (6) (1990) 1019–1033.
[8] P. Jaillet, The probabilistic vehicle routing problem, Vehicle Routing Methods Stud. (1988) 293–318.
[9] P. Jaillet, Stochastic routing problems, Stoch. Combin. Optim. (1987) 178–186.
[10] X. Li, P. Tian, S.C. Leung, Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm, Int. J. Prod. Econ. 125 (1) (2010) 137–145.
[11] J. Zhang, W.H. Lam, B.Y. Chen, A stochastic vehicle routing problem with travel time uncertainty: trade-off between cost and customer service, Netw. Spat. Econ. 13 (4) (2013) 471–496.
[12] M. Dror, M. Ball, B. Golden, A computational comparison of algorithms for the inventory routing problem, Ann. Oper. Res. 4 (1) (1985) 1–23.
[13] R.C. Larson, Transporting sludge to the 106-mile site: An inventory/routing model for fleet sizing and logistics system design, Transp. Sci. 22 (3) (1988) 186–198.
[14] V. Lambert, G. Laporte, F. Louveaux, Designing collection routes through bank branches, Comput. Oper. Res. 20 (7) (1993) 783–791.
[15] M. Dror, P. Trudeau, Stochastic vehicle routing with modified savings algorithm, European J. Oper. Res. 23 (2) (1986) 228–235.
[16] D. Teodorovic, P. Lucic, Intelligent vehicle routing system, in: ITSC2000. 2000 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 00TH8493), IEEE, 2000, pp. 482–487.
[17] N. Secomandi, F. Margot, Reoptimization approaches for the vehicle-routing problem with stochastic demands, Oper. Res. 57 (1) (2009) 214–230.
[18] C. Novoa, R. Storer, An approximate dynamic programming approach for the vehicle routing problem with stochastic demands, European J. Oper. Res. 196 (2) (2009) 509–515.
[19] M. Nazari, A. Oroojlooy, L.V. Snyder, M. Takáč, Reinforcement learning for solving the vehicle routing problem, 2018, pp. 1–21, ArXiv preprint arXiv:1802.04240.
[20] A.M. Florio, R.F. Hartl, S. Minner, Optimal a priori tour and restocking policy for the single-vehicle routing problem with stochastic demands, European J. Oper. Res. 285 (1) (2020) 172–182.
[21] M.H. Alrefaei, S. Andradóttir, A simulated annealing algorithm with constant temperature for discrete stochastic optimization, Manage. Sci. 45 (5) (1999) 748–764.
[22] F. Glover, M. Laguna, R. Marti, Principles of Tabu Search in Approximation Algorithms and Metaheuristics, Chapman and Hall/CRC, 2005.
[23] S.R. Thangiah, Vehicle Routing with Time Windows using Genetic Algorithms, Citeseer, 1993.
[24] E. Bonabeau, D.d.R.D.F. Marco, M. Dorigo, G. Théraulaz, G. Theraulaz, et al., Swarm Intelligence: from Natural to Artificial Systems, vol. 1, Oxford university press, 1999.
[25] L.N. Castro, L.N. De Castro, J. Timmis, Artificial Immune Systems: a New Computational Intelligence Approach, Springer Science & Business Media, 2002.
[26] N. Gunantara, A review of multi-objective optimization: Methods and its applications, Cogent Eng. 5 (1) (2018) 1502242.
[27] R. Ojstersek, B. Miran, B. Borut, Multi-objective optimization of production scheduling with evolutionary computation: A review, Int. J. Ind. Eng. Comput. 11 (3) (2020) 359–376.
[28] B. Xue, M. Zhang, W.N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, IEEE Trans. Cybern. 43 (6) (2012) 1656–1671.
[29] Y. Xue, B. Xue, M. Zhang, Self-adaptive particle swarm optimization for large-scale feature selection in classification, ACM Trans. Knowl. Discov. Data 13 (5) (2019) 1–27.
[30] Y. Zhang, D.-w. Gong, J. Cheng, Multi-objective particle swarm optimization approach for cost-based feature selection in classification, IEEE/ACM Trans. Comput. Biol. Bioinform. 14 (1) (2015) 64–75.
[31] Y. Xue, Y. Tang, X. Xu, J. Liang, F. Neri, Multi-objective feature selection with missing data in classification, IEEE Trans. Emerg. Topics Comput. Intell. (2021) 1–10.
[32] K.C. Tan, C.Y. Cheong, C.K. Goh, Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation, European J. Oper. Res. 177 (2) (2007) 813–839.
[33] S.B. Gee, W.A. Arokiasami, J. Jiang, K.C. Tan, Decomposition-based multi-objective evolutionary algorithm for vehicle routing problem with stochastic demands, Soft Comput. 20 (9) (2016) 3443–3453.
[34] R.S. Michalski, Learnable evolution model: Evolutionary processes guided by machine learning, Mach. Learn. 38 (1) (2000) 9–40.
[35] B. Moradi, The new optimization algorithm for the vehicle routing problem with time windows using multi-objective discrete learnable evolution model, Soft Comput. (2019) 1–29.
[36] X. Zhang, Y. Tian, R. Cheng, Y. Jin, An efficient approach to nondominated sorting for evolutionary multiobjective optimization, IEEE Trans. Evol. Comput. 19 (2) (2014) 201–213.
[37] M.M. Solomon, Algorithms for the vehicle routing and scheduling problems with time window constraints, Oper. Res. 35 (2) (1987) 254–265.
[38] C.M. Fonseca, L. Paquete, M. López-Ibánez, An improved dimension-sweep algorithm for the hypervolume indicator, in: 2006 IEEE International Conference on Evolutionary Computation, IEEE, 2006, pp. 1157–1163.
[39] J.R. Schott, Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization (Ph.D. thesis), Massachusetts Institute of Technology, 1995.
[40] D.A. Van, V. Gary, B. Lamont, Multiobjective evolutionary algorithm research: A history and analysis, Evol. Comput. 8 (2) (1998) 1–88.
[41] J. Li, L. Xin, Z. Cao, A. Lim, W. Song, J. Zhang, Heterogeneous attentions for solving pickup and delivery problem via deep reinforcement learning, IEEE Trans. Intell. Transp. Syst. (2021) 1–10.
[42] L. Xin, W. Song, Z. Cao, J. Zhang, Step-wise deep learning models for solving routing problems, IEEE Trans. Ind. Inf. 17 (7) (2020) 4861–4871.
[43] Y. Wu, W. Song, Z. Cao, J. Zhang, A. Lim, Learning improvement heuristics for solving routing problems.., IEEE Trans. Neural Netw. Learn. Syst. (2021) 1–10.