APEM journal
Advances in Production Engineering & Management
Volume 14 | Number 2 | June 2019 | pp 201–212
https://doi.org/10.14743/apem2019.2.322
ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

Time-dependent and bi-objective vehicle routing problem with time windows

Zhao, P.X.[a,*], **Luo, W.H.**[a], **Han, X.**[a]

[a]School of Management, Shandong University, Jinan, Shandong, P.R. China

ABSTRACT

The optimization of bi-objective vehicle routing problem has become a research hotspot in recent years. In this paper, a time-dependent and bi-objective vehicle routing problem with time windows (TD-BO-VRPTW) is proposed, which is a new extension of classical vehicle routing problem. Time-dependency is presented for the situation that vehicle's travel speed is affected by its departure time and the distance between two customers. The total transportation costs and time costs are two objectives optimized simultaneously through constructing a bi-objective mixed integer linear programming model. To deal with this problem, the non-dominated sorting genetic algorithm II (NSGA-II) is adopted to obtain the Pareto optimal solution set. In the numerical examples, the RC108 from Solomon's benchmark set is employed and the results in the Pareto front show the efficiency of NSGA-II for the TD-BO-VRPTW. To further test the performance of this algorithm, two objectives are optimized separately and then the sum of two objectives is also optimized. Through comparing these results with solutions in the Pareto front, it can be concluded that the algorithm is reliable, and the results in Pareto front are competitive because there is a trade-off between two objectives.

ARTICLE INFO

Keywords:
Vehicle routing problem;
Time-dependency;
Bi-objective optimization;
Time windows;
Pareto optimal solutions;
Evolutionary algorithms;
NSGA-II algorithm

**Corresponding author:*
pxzhao@sdu.edu.cn
(Zhao, P.X.)

Article history:
Received 14 February 2018
Revised 10 May 2019
Accepted 17 May 2019

1. Introduction

The Vehicle Routing Problem (VRP) was first proposed by Dantzig and Ramser in 1959 [1]. It is defined on a network, in which the points represent the depot and customers and the weights on the arcs stand for the distance between two points. The depot is the start point and end point of each route, and each customer has a demand that must be served by a vehicle. Each vehicle serves the customers along its route and the sum of all demands on this route cannot exceed the maximum capacity. VRP aims to find the optimal solution and thus the objective function is optimized and the corresponding constraints are met when each vehicle travels along its own route. Zhu *et al*. proposed an improved hybrid algorithm based on heuristic to solve VRP under multi-depots condition [2]. Chiang and Hsu considered that VRP combines the characteristics of two typical NP-hard optimization problems including traveling salesman problem (TSP) and bin packing problem (BPP) [3]. Similar to VRP, TSP aims to find the optimal route, but it only considers the situation when the number of vehicles is equal to one**.** BPP assigns customers to different vehicles to minimize the number of vehicles and satisfies all the constraints at the same time, but it does not consider the customer service priorities.

The vehicle routing problem with time windows (VRPTW), which was first introduced by Solomon in 1987, is an extension of VRP [4]. In the VRPTW, each customer is assigned with a time window who must be served within the time window. Cao *et al*. solved the vehicle routing

problem with multiple fuzzy time windows using an improved wolf pack algorithm [5]. Wu *et al*. proposed a co-evolutionary based algorithm and applied it to solve VRPTW [6]. Yu *et al*. proposed an improved branch-and-price (BAP) to handle the VRPTW with heterogeneous fleet [7]. Miranda and Conceição proposed an extension of VRPTW in which the travel time and service time are stochastic and solved it by metaheuristic [8]. In the present study, we regard the time window as a soft constraint. If the vehicle arrives outside the time window, no matter earlier or later, it should pay for the penalty cost. Many real-world applications can be considered as the VRPTW, such as postal delivery, train and bus scheduling as well as waste collection. VRPTW is a typical NP-hard combinatorial optimization problem. Due to its practical value and high computational complexity, it has become a research hotspot in operational research and management science.

The bi-objective vehicle routing problem (BOVRP) is another extension of VRP. The classical way to solve BOVRP is to add the two objectives together and solve it as a single-objective optimization problem. This method is essentially to search for a single-objective optimal solution. Another way is to find the Pareto optimal solution set, which is a trade-off between two objectives. In this set, solutions are not going to be worse than any other solutions on both objective function values simultaneously. Geiger was one of the first researchers solving multi-objective vehicle routing problem by using Pareto approaches [9]. Then Barán and Schaerer proposed a multi-objective ant colony algorithm to minimize the number of vehicles, travel distance and service time simultaneously [10]. Tan *et al*. put forward a hybrid multi-objective evolutionary algorithm (HMOEA) to deal with BOVRP, in which the two objectives are number of vehicles and travel distance [11]. Ombuki *et al*. designed a multi-objective genetic algorithm to minimize the number of vehicles and travel distance [12]. Garcia-Najera and Bullinaria proposed a multi-objective evolutionary algorithm (MOEA), which is characterized by considering individual similarity in the selection process [13]. Chiang and Hsu proposed a knowledge-based evolutionary algorithm to solve bi-objective vehicle routing problem with time windows, minimizing the number of vehicles and travel distance [3]. Qi *et al*. proposed a memetic algorithm based on decomposition and applied it to solve multi-objective VRPTW [14]. Iqbal *et al*. proposed a new model for multi-objective VRP, and solved it on the basis of local search [15]. Some researchers considered the minimization of time window constraint violations as an objective. For example, Xu *et al*. optimized the constraint violation and two other objectives by an Or-opt NSGA-II [16]. Castro *et al*. considered the constraint violation of vehicle capacity and time window as two objectives [17].

The time-dependent vehicle routing problem (TDVRP) was initially proposed by Malandraki and Daskin to capture the congestion in a traffic network [18]. Hill and Benton constructed a model for time-dependent conditions, which lays the foundation of many studies [19]. Malandraki and Dial solved TSP under time-dependent conditions by dynamic programming [20]. However, all of the above studies disrespect the First-In-First-Out (FIFO) principle, indicating that it is possible that a vehicle departing later arrives earlier. Ichoua *et al*. proposed a time-dependent model in which travel speed is a step function and travel time is a piecewise linear function [21]. Fleishmann *et al*. proposed a method to construct time-dependent model in which the travel time was smoothed by a step function [22]. These methods respect the FIFO principle. These two methods are widely used in studies afterwards. Then, Huang *et al*. proposed the conception of path flexibility for TDVRP and solved it by CPLEX [23]. Çimen and Soysal extended the conception of TDVRP to stochastic conditions and proposed a heuristic to solve it [24].

Based on previous research, we combine the bi-objective optimization, the time-dependent travel time model and the time window as well as innovatively propose the time-dependent and bi-objective vehicle routing problem with time windows (TD-BO-VRPTW). In the remainder of this paper, we first give a detailed description of the TD-BO-VRPTW in Section 2. In Section 3, we introduce the notion of NSGA-II. Our main results are given in Section 4. Finally, Section 5 contains a brief summary.

2. Problem description and model construction

This section mainly includes the notion of time-dependent travel time model, the definition of parameters and decision variables, assumptions and mathematical model.

2.1 Time-dependent travel time model

As mentioned previously, Malandraki and Daskin first proposed the conception of time-dependent travel time model [18]. Fig. 1 shows the relationship between travel time and departure time when the distance is 1, presenting that the travel time between two points is a step function of departure time. The main feature of this model is that it does not respect the FIFO principle. For example, a vehicle leaving at $t_1 = 1$ will arrive at $t_2 = 4$, while a vehicle leaving at $t_3 = 2$ will arrive at $t_4 = 3$. Hill and Benton proposed another time-dependent model in which travel speed is a step function of the departure time, as shown in Fig. 2. This model still disrespects the FIFO principle [19]. In order to give a better description of time-dependency, adjustment of travel speed should be taken into account when vehicle travels beyond the bounds of the time intervals. Ichoua *et al*. proposed a time-dependent travel time model in which travel speed is a step function and travel time is a piecewise linear function, as shown in Fig. 3 and Fig. 4 [20].

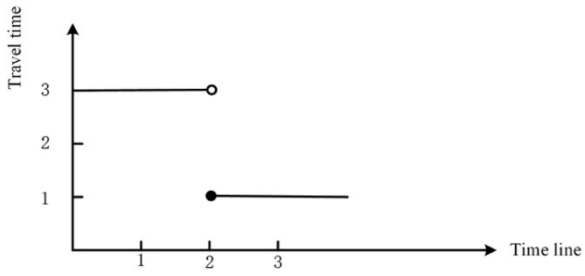

Fig. 1 Time-dependent model I

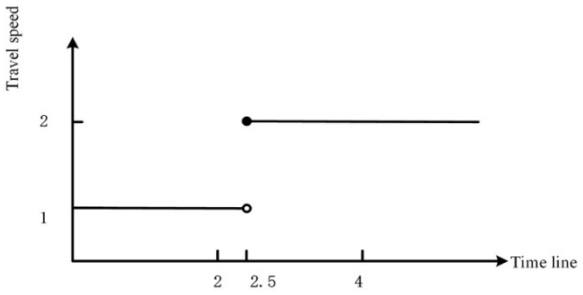

Fig. 2 Time-dependent model II

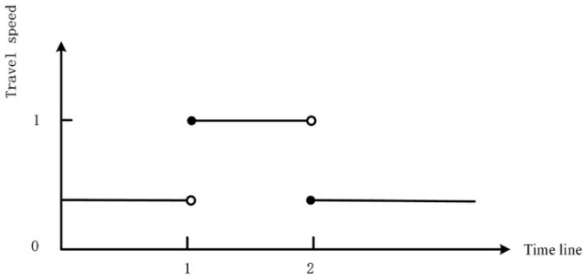

Fig. 3 Time-dependent model III: Speed

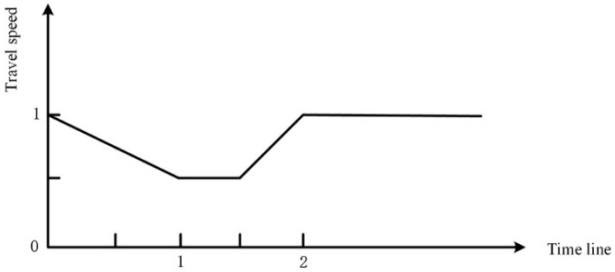

Fig. 4 Time-dependent model III: Time

Fig. 3 shows the relationship between travel speed and departure time, and Fig. 4. provides an example of travel time function for an arc of length 1. For a non-time-dependent and symmetrical road model, travel time of an arc (v_i, v_j) can be simply expressed as $t_{ij} = d_{ij}/v_{ij}$, where v_{ij} is the average travel speed on arc (v_i, v_j) and d_{ij} is the distance of arc (v_i, v_j). However, in time-dependent model, travel speed changes with departure time. Suppose the departure time is t_0, and travel speed is a step function of departure time, wherein speed of the k-th time period is v_{ij}^k, corresponding to the departure time interval for $[t_k, t_{k+1}]$. Then, the process of calculation is expressed as follows:

Step 1: Calculate the value of k according to $t_0 \in [t_k, t_{k+1}]$, and get the corresponding v_{ij}^k, assign $d_{ij} \to d$, calculate $t' = t_0 + d/v_{ij}^k$.

Step 2: If $t' \leq t_{k+1}$, then output arrival time t' and travel time $t' - t_0$, end the loop. Otherwise, go to Step3.

Step 3: Update $d_{ij} - v_{ij}^{k}(t_{k+1} - t_0) \rightarrow d$, calculate $t' = t_{k+1} + d/v_{ij}^{k+1}$,update $k+1 \rightarrow k$, return to Step 2.

This time-dependent model respects the FIFO principle. Compared with the former two time-dependent models, it can better capture congestion in a traffic network. Therefore, we choose this model for further study.

2.2 Assumptions and symbolic descriptions

The following assumptions will be needed throughout the paper:

- Travel speed is variable, and there are multiple time intervals. Speed in a certain time interval is constant.
- Each customer is assigned with a time window and vehicles arriving in advance or beyond the deadline must pay for penalty cost.
- Each customer must be served exactly once by one vehicle.
- Each vehicle must depart from the depot and return to the depot after completing the delivery task.
- Vehicles are homogeneous, and the constraints of maximum driving distance and maximum driving time are not considered.

Based on the above assumptions, the present study aims to minimize transportation costs and time costs. Transportation costs include depreciation costs and fuel costs, and time costs include waiting costs and penalty costs. Since the problem involved in the current work is essentially a derivative model of classical VRPTW, it can be described by the traditional network fluid formula system as follows [25]:

Let $G = (V, D)$ represents a complete graph, where V= $\{v_0,v_1,\ldots,v_{n+1}\}$ is the point set. Therefore, the subset C=$\{v_1,v_2,\ldots,v_{n+1}\}$ are the customers. Then, define the node sets V_0=$\{v_0,v_1,\ldots,v_n\}$ and V_{n+1}=$\{v_1,v_2,\ldots,v_{n+1}\}$, and arc set D=$\{(v_i, v_j)|\ (v_i, v_j)|v_i \in V_0, v_j \in V_{n+1}, i \neq j\}$. Each vehicle $k \in K$ has a maximum capacity Q. Besides, each customer $i \in C$ has a demand $q_i \geq 0$, a service time $s_i \geq 0$ and a time window $[e_i, l_i]$. The depot also has a time window $[0, T_{max}]$. The distance of arc $(v_i, v_j) \in D$ is d_{ij}. c^f is the vehicle's fuel cost per unit distance, and c^d is the vehicle's depreciation cost. p_e is the waiting cost per unit time, and p_l is the penalty cost per unit time.

2.3 Mathematical model

First, we define two decision variables:

- Binary decision variable $x_{ij}^{k}, \forall k \in K, \forall i \in V_0, \forall j \in V_{n+1}$.

$$x_{ij}^{k} = \begin{cases} 1 \text{ If vehicle } k \text{ travels from customer } i \text{ to customer } j \\ 0 \text{ Otherwise} \end{cases}$$

- Non-negative real decision variable y_i^k: The departure time after vehicle k has served customer i.

According to the previous time-dependent model, travel speed on arc (v_i, v_j) is a function of the departure time $v_{ij}(y_i)$ and then travel time can be denoted by d_{ij} and $v_{ij}(y_i)$, i.e. $t_{ij}(d_{ij}, v_{ij}(y_i))$. Travel speed function $v_{ij}(y_i)$ and travel time function $t_{ij}(d_{ij}, v_{ij}(y_i))$ are a step function and a piecewise linear function, respectively. Based on this notation, the TD-BO-VRPTW can be constructed as the following bi-objective mixed integer linear programming model:
Objective functions:

$$minf_1 = \sum_i \sum_j \sum_k c^f x_{ij}^{k} d_{ij} + c^d \sum_k \sum_j x_{0j}^{k} \tag{1}$$

$$minf_2 = p_e \sum_i max(e_i - y_i^k + s_i, 0) + p_l \sum_i max(y_i^k - s_i - l_i, 0) \tag{2}$$

Constrains:

$$\sum_{i\in C}\sum_{j\in V_{n+1}} q_i x_{ij}^k \le Q, \forall k \in K \tag{3}$$

$$\sum_{k\in K}\sum_{j\in V_{n+1}} x_{ij}^k = 1, \forall\, i \in C \tag{4}$$

$$\sum_{i\in V_0} x_{ih}^k - \sum_{j\in V_{n+1}} x_{hj}^k = 0, \forall h \in C, \forall k \in K \tag{5}$$

$$x_{i0}^k = x_{n+1,i}^k = 0, \forall i \in C, \forall k \in K \tag{6}$$

$$\sum_{i\in C} x_{0i}^k = \sum_{i\in C} x_{i,n+1}^k \le 1, \forall k \in K \tag{7}$$

$$\left(y_i^k + t_{ij}\left(d_{ij}, v_{ij}\left(y_i^k\right)\right)\right) x_{ij}^k \le y_j^k - s_j, \forall i \in V_0, \forall j \in V_{n+1} \forall k \in K \tag{8}$$

$$x_{ij}^k \in \{0,1\}, \forall i \in V_0, \forall j \in V_{n+1} \forall k \in K \tag{9}$$

$$y_i^k \ge 0, \forall i \in V, \forall k \in K \tag{10}$$

The objective function (Eq. 1) minimizes the transportation costs, and the objective function (Eq. 2) minimizes the time costs. Expressions (Eq. 3) are the maximum capacity constraints, indicating that for each route, the sum of all demands cannot exceed the maximum capacity. Constraints (Eq. 4) ensure that each customer is visited exactly once by one vehicle. Constraints (Eq. 5) and (Eq. 6) are the vehicle flow conservation constraints. Constraints (Eq. 7) stipulate that each vehicle can be used at most once. Expressions (Eq. 8) are the arrival time constraint, indicating that a vehicle must arrive at the next customer earlier than its service start time. Expressions (Eq. 9) and (Eq. 10) are variable specification constraints.

3. Used methods

Recently, the non-dominated sorting genetic algorithm II (NSGA-II) proposed by Deb has been extensively responded and applied [26]. The selection process of individuals is improved by employing the elite strategy, density value estimation strategy and fast non-dominated sorting strategy. This genetic algorithm has been widely used in various fields. It not only ensures the value of the optimal solution and obtains the Pareto front, but also reduces the algorithm complexity and ensures the efficiency of the algorithm. It has become a classical algorithm in the field of multi-objective optimization. Therefore, this algorithm is adopted to solve the problem. This section mainly introduces integer linear programming, NSGA-II and its related conceptions.

3.1 Introduction to integer linear programming and NSGA-II

Integer linear programming indicates that the values of some or all decision variables in linear programming are integers. Integer linear programming is mostly NP-hard. Therefore, when the scale of the problem is large, it will be difficult to solve the problem. Integer linear programming is an important issue in operations research. In recent decades, numerous scholars have studied the solution to solve this problem. The most commonly used method is the heuristic algorithm, whose principle is to find a feasible solution within an acceptable range. The advantage is that the convergence speed is fast, and the disadvantage is that it may fall into local optimum. Common heuristic algorithms include genetic algorithm, tabu search, simulated annealing, etc.

The NSGA-II algorithm was proposed based on genetic algorithm. The non-dominated sorting approach is put forward in order to give every solution in the population a non-domination rank.

The non-domination rank starts at 1 and increases by 1 each round. In each round of the loop, every unsorted individual p in the population is compared with all the remaining unsorted individuals to determine whether the individual p dominates all the other unsorted individuals. If so, individual p is assigned with the current non-domination rank. Finally, all individuals in the population are assigned with a non-domination rank.

Additionally, crowding distance is also defined in NSGA-II to get an estimate of the density of solutions surrounding a particular solution. In terms of bi-objective optimization, calculate the average distance of two points on either side of a particular point along each objective. This value can be regarded as the sum of the adjacent two sides of a rectangle formed by two adjacent solutions on both sides of a particular solution as vertices. As presented in Fig. 5, the crowding distance of the th i-th solution is equal to half the perimeter of the rectangle formed by its two adjacent solutions as vertices, as shown by dashed line.

Each solution is assigned with a crowding distance by calculation. For individuals with the same non-dominated rank, the larger the crowding distance, the better the diversity. Therefore, it should be preferred in the selection process. Based on the definition of non-domination rank i_{rank} and crowding distance $i_{distance}$, two solutions i and j in the population can be compared using the comparison operator:

$i \prec j$
if: $i_{rank} < j_{rank}$
or: $i_{rank} = j_{rank}$ and $i_{distance} > j_{distance}$

That is, in the selection process of the algorithm, it is preferred when a solution has a lower non-domination rank. Otherwise when the two solutions have the same non-dominated rank, we prefer the solution with larger crowding distance due to its better diversity.

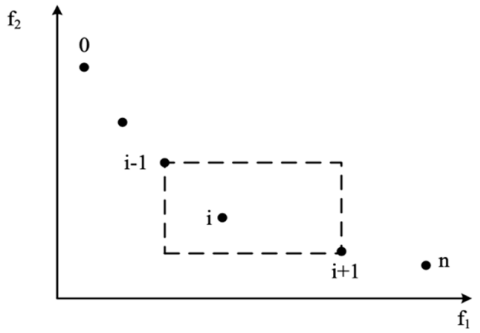

Fig. 5 Crowding distance

3.2 Main loop

The main loop of NSGA-II can be found in Fig. 6.

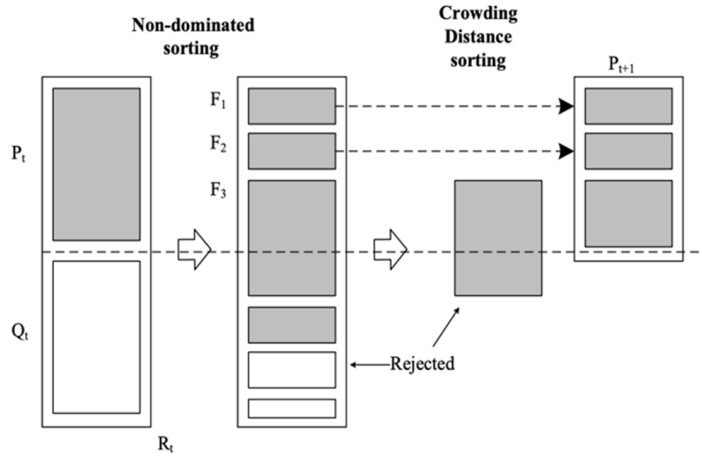

Fig. 6 Main loop

First, generate the initial population P_0, which contains N solutions. Then, the population is sorted based on domination, and each solution is assigned with a non-domination rank. An offspring population Q_0 of size N is created by selection, crossover and mutation operators. These two populations are combined to get $R_0 = P_0 \cup Q_0$. Finally, the population R_0 is trimmed based on non-domination rank and crowding distance to ensure that the population is the same size at the beginning of each iteration.

4. Results and discussion

Solomon's benchmark set is widely used in VRP, which consists of 56 instances categorized into 6 sets. Each instance includes 1 depot and 100 customers. According to the spatial distribution of customers, these instances can be divided into 3 categories including C (clustered), R (random) and RC (mixed). It is assumed that the travel speed of all arcs conforms to a unified time-dependent step function. The time window of depot $[0, T_{max}]$ ($T_{max} = 240$) is divided into five intervals on average, and the average speed of each interval is [1, 1.6, 1.05, 1.6, 1]. In this paper, RC108 is selected to test the performance of the algorithm. In this instance, customers are clustered and randomly distributed, the maximum capacity of vehicle Q is 200, and the service time s_i is 10.

All the experiments are based on MATLAB R2017a, and the algorithm parameters are set as follows: Population size $Popsize = 100$, maximum number of iterations $IterMax = 500$, crossover probability $p_c = 0.5$ and mutation probability $p_m = 0.1$. Vehicle's fuel cost per unit tance $c^f = 0.5$, depreciation cost $c^d = 50$, the time cost coefficient is $p_e = 0.5$ and $p_l = 5$. Fig. 7 represents the first non-dominated front, namely Pareto front. There are 27 solutions in the Pareto optimal set. Table 1 shows the detailed data.

The results in Fig. 7 show that these two objectives f_1 and f_2 are not positively correlated. If the two objectives change in the same direction, some solutions will be better on both objectives and thus the number of Pareto solutions becomes small. There are 27 solutions in Pareto front, indicating that the two objectives are contradictory to some extent. For one objective to become better, it is necessary to take the other to become worse.

To further compare the performance of the algorithm, we use single-objective optimization algorithm to optimize f_1 and f_2 respectively. We use the classical genetic algorithm to carry out this experiment. The relevant parameters are set as before. First, f_1 is optimized and the final optimization result is shown in Fig. 8. The left figure represents the optimal routes, where the square and circles represent the depot and customers respectively, and the right figure stands for the change of the objective function f_1 with the number of iterations. Fig. 9 is obtained by optimizing f_2. The detailed route of each vehicle and the values of f_1 and f_2 are shown in Table 2 and Table 3. Finally, f_1 and f_2 are added together as one objective and optimized. The results are shown in Table 4 and Fig. 10.

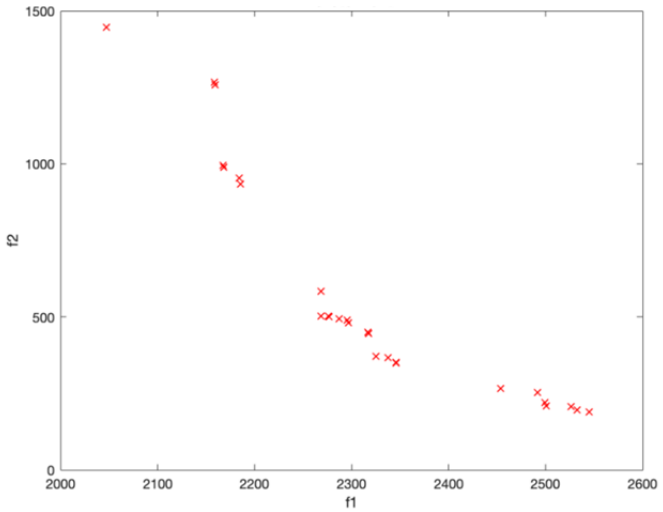

Fig. 7 Pareto front

Table 1 Detailed data of Pareto front

Solution No.	Values of Pareto optimal solutions	
	f_1	f_2
1	2047.2581	1445.5884
2	2158.6574	1266.6068
3	2159.2772	1258.6648
4	2167.7613	996.1304
5	2168.3811	988.1884
6	2184.0555	954.5300
7	2185.5761	934.0495
8	2268.4119	583.7399
9	2268.8158	502.9485
10	2276.2644	502.6524
11	2276.8878	499.6613
12	2287.2728	494.8850
13	2295.2586	489.3503
14	2296.7619	480.1851
15	2316.6997	450.1220
16	2317.8669	444.9339
17	2325.2467	370.6427
18	2337.3527	367.2576
19	2345.8875	352.2281
20	2345.8913	348.4370
21	2453.7982	265.2944
22	2492.1418	253.8482
23	2499.1596	220.4622
24	2500.9599	208.9021
25	2526.6566	207.1993
26	2532.5440	195.3103
27	2545.1208	190.0310

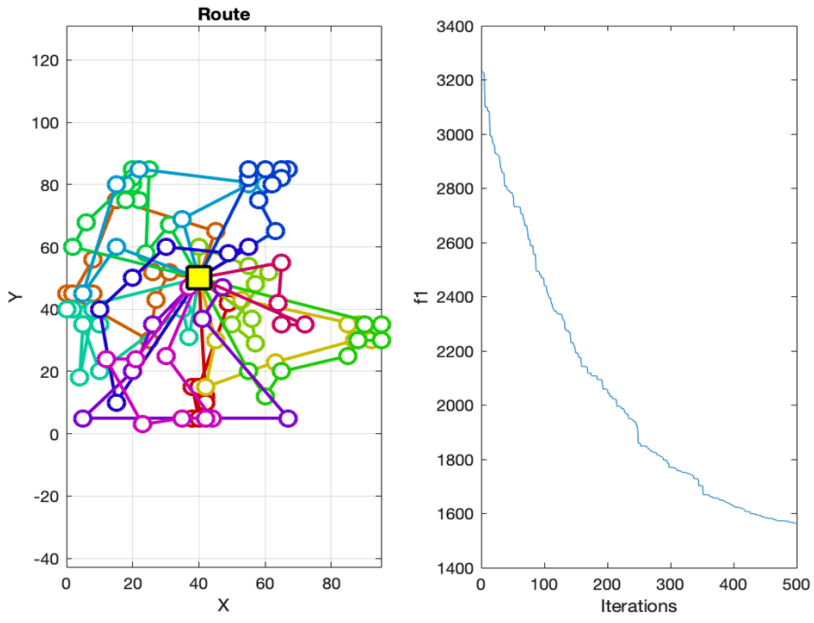

Fig. 8 Results when optimizing f_1

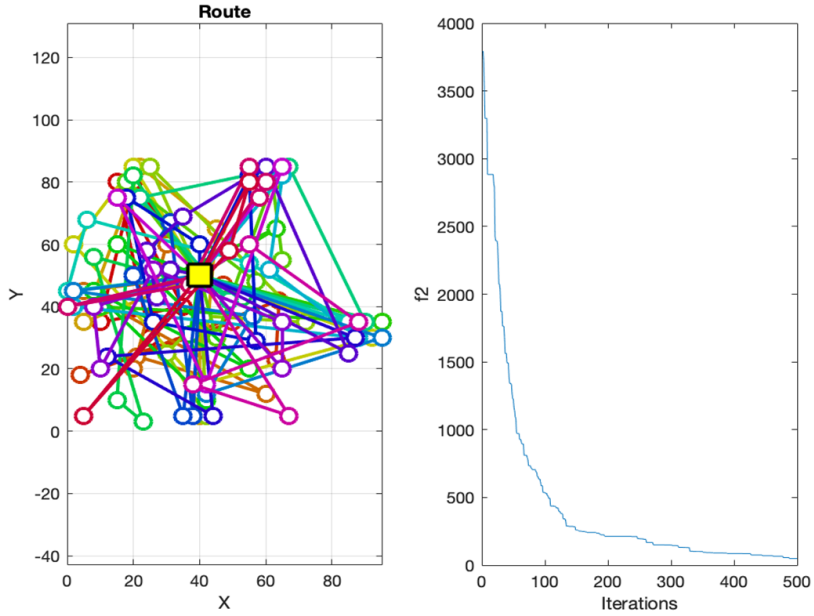

Fig. 9 Results when optimizing f_2

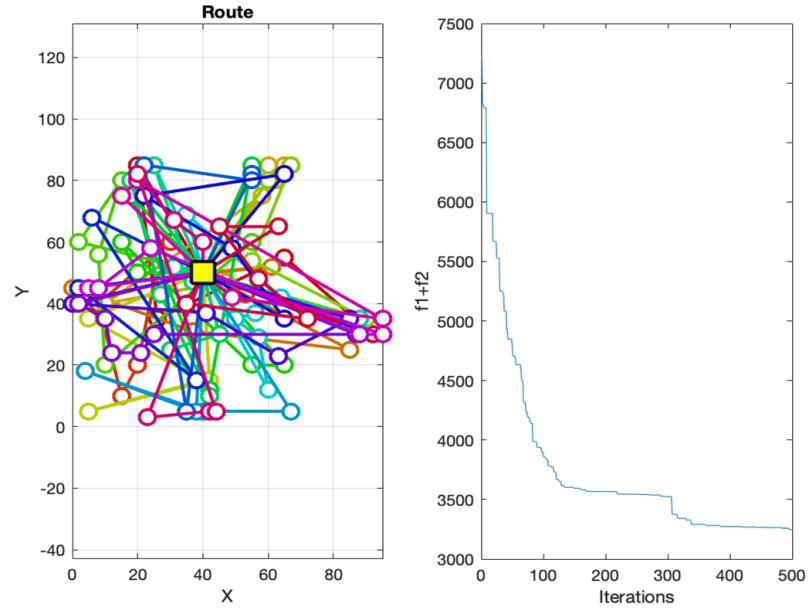

Fig. 10 Results when optimizing $f_1 + f_2$

Table 2 Detailed data of optimal solution when optimizing f_1

Vehicle No.	Optimal routes and the value of f_1 and f_2 when optimizing f_1
1	0-49-19-23-21-24-22-91-0
2	0-98-69-82-52-17-12-47-78-7-61-0
3	0-92-31-34-28-32-85-20-64-0
4	0-68-96-94-93-56-84-95-0
5	0-51-76-63-33-30-26-29-27-0
6	0-100-88-1-5-4-45-2-6-46-79-73-0
7	0-15-16-9-11-97-13-59-65-83-0
8	0-60-14-8-3-39-42-70-0
9	0-44-43-40-35-36-37-38-41-72-0
10	0-58-10-53-55-81-54-0
11	0-99-74-75-89-66-80-0
12	0-57-18-48-25-77-87-86-90-0
13	0-50-62-67-71-0
f_1	1564.0496
f_2	3909.1340

Table 3 Detailed data of optimal solution when optimizing f_2

Vehicle No.	Optimal routes and the value of f_1 and f_2 when optimizing f_2
1	0-52-8-65-9-4-0
2	0-85-67-97-56-80-0
3	0-14-76-86-74-55-0
4	0-21-61-3-13-0
5	0-5-73-22-28-0
6	0-48-57-66-50-1-0
7	0-64-38-71-94-46-0
8	0-27-10-72-0
9	0-83-19-12-51-60-0
10	0-92-78-77-58-45-0
11	0-29-35-2-91-0
12	0-17-79-96-34-0
13	0-15-30-93-37-0
14	0-47-95-49-26-0
15	0-23-100-53-25-0
16	0-44-84-99-6-68-0
17	0-18-87-32-0
18	0-69-82-88-70-40-33-0
19	0-11-59-98-62-63-0
20	0-7-20-36-0
21	0-89-24-31-54-0
22	0-16-90-43-41-39-0
23	0-75-81-42-0
f_1	3572.6748
f_2	48.6028

Table 4 Detailed data of optimal solution when optimizing $f_1 + f_2$

Vehicle No.	Optimal routes and the value of f_1 and f_2 when optimizing $f_1 + f_2$
1	0-28-71-99-5-0
2	0-92-98-74-58-11-55-0
3	0-33-83-17-93-0
4	0-36-38-40-41-10-80-0
5	0-13-22-75-20-0
6	0-39-35-96-29-0
7	0-73-8-78-59-54-0
8	0-53-60-51-63-0
9	0-6-49-43-0
10	0-64-57-19-82-46-0
11	0-31-69-90-1-0
12	0-76-84-56-95-67-70-0
13	0-89-21-97-23-0
14	0-44-42-3-25-0
15	0-79-24-47-0
16	0-62-81-2-37-0
17	0-16-66-85-34-0
18	0-15-9-87-86-52-32-0
19	0-14-12-88-30-26-0
20	0-7-4-27-91-0
21	0-77-48-18-45-100-68-0
22	0-72-61-94-50-65-0
f_1	3115.7205
f_2	132.0221

According to Table 1, Table 2 and Table 3, when f_1 is optimized, the optimal solution $f_1 = 1564.0496$ and the corresponding $f_2 = 3909.1340$; when f_2 is optimized, the result is $f_1 = 3572.6748$ and $f_2 = 48.6028$. Obviously, when optimization is carried out with one objective, the other objective will be significantly worse, which is consistent with the previous analysis that f_1 and f_2 are contradictory. For example, when f_1 is optimized, the value of optimal solution is 1564.0496, which is better than all the solutions in Pareto front. However, the corresponding f_2 is 3909.1340, which is worse than the worst in Pareto front, i.e. 1445.5884. The same goes for the situation when f_2 is optimized. From Fig. 8 and Fig. 9, we can also find that when f_1 is opti-