Team Control Number

20200926755

Problem Chosen

C

2020

ShuWei Cup

Summary

Based on the tasks and road distribution of snow cleaning system in urban areas, this paper constructs a model to analyze the road selection of the cleaning vehicle lane, and then discusses the operation scheme of the cleaning vehicle operating system under the shortest total distance and the shortest time. Then, combined with relevant constraints, the optimal path plan is given.

To solve the first problem, based on node location and connection table, the connection relation between each point in the problem and its relation needs to be converted into the weighted direct graph in mathematics before the model is established. As the network is relatively dense, Floyd algorithm is used to calculate the weight of each side, which is the distance weight between the two connection points. Considering that the width of each road is different, a certain width weight is added to the distance data of each road according to the width of the road to obtain the final distance weight. Combining the objective functions and constraint conditions, and using Dijkstra and Genetic algorithm to solve the shortest total mileage in Matlab. The shortest total mileage in this case is 2178.1347km.

For question two, in the case of a large amount of snow, there will be snow piling up points, then snowplows, trucks and sanitation workers for joint operations are needed, based on the optimal path of the vehicle in question one. According to the optimal path constraint conditions of the transport vehicle, a new objective function is established. Using Dijkstra and genetic algorithm to solve the shortest total mileage in Matlab, the shortest total mileage in this case is 2746.0036km.

For questions three and four, both of them are based on question one, combined with the priority of the demand arc (the priority of the road), and establish a new mathematical model based on the new objective function and constraint conditions. According to the complexity of variables and constraints, a new heuristic algorithm based on parallel and genetic algorithms is established, then, using the whole process to find the best way to solve the shortest time required for cleaning. The shortest time to get the third question is 6.99h and the shortest time for question four is 5.92h.

Key word: Optimal path; Floyd algorithm; Dijkstra algorithm; Genetic algorithm; Parallel Computing

Content

1. Introduction

1.1 Background

Heading into winter, as the weather gets colder, massive snow has fallen in northern China, and it is likely to intensify in the future. The heavy snowfall has caused severe road congestion and seriously affected the normal life of residents, leading to the suspension of work and school in some cities. The sanitation workers raced against time to clear the snow and get the road clear as soon as possible. Therefore, in order to realize the smooth road, it is urgent to have a reasonable road snow removal scheme.

1.2 Work

Task 1: Under the assumption that the snow can be swept to the nearby green belt or leisure area and the number of vehicles for cleaning is fixed, provide a reasonable cleaning scheme.

Task 2: In the case of a large amount of snow and the municipal sanitation workers have relatively fixed configurations for snowplow, transporter and sanitation workers, provide the optimal work plan for snowplow, transporter and sanitation workers that can accomplish the task of snow shoveling the fastest.

Task 3: Provide a reasonable snow removal schedule, so as to clear the vehicles parked on both sides of the road in advance and restore the use of parking Spaces as soon as possible.

Task 4: Where different roads have different cleaning priorities, provide an optimal snow removal plan that takes into account road priorities.

2. Problem analysis

Snow removal of urban roads is a very complex system, and the optimization of its mechanism and mode affects the operation and development of cities. In this part, we will analyze the problems to provide ideas for the later research.

2.1 Data analysis

First, we preprocessed the data. According to the road information of a city given in the question, including the longitude and latitude of the intersection, the connection relationship of the intersection, the width of the intersection and the location of potential snow accumulation. Matlab software was used to process the longitude and latitude of the intersection, turning it into Cartesian coordinate points, as shown in Figure 2-1.

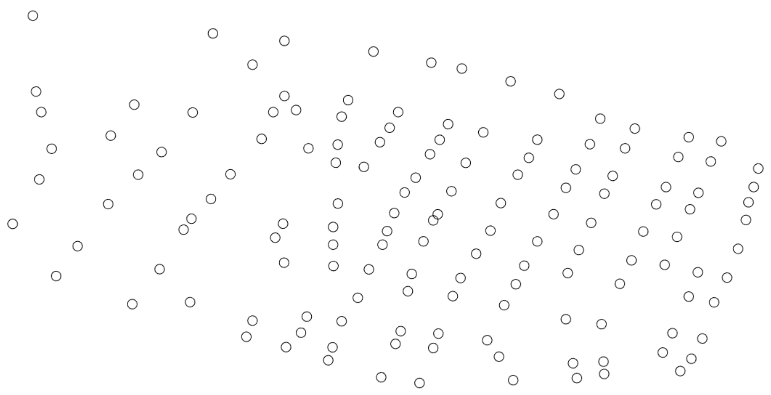
Figure 2-1 Road intersection distribution map

Then, we processed the connection relation of the intersection and the width of the intersection (i.e. the width of the two intersections) is obtained as shown in Figure 2-2.

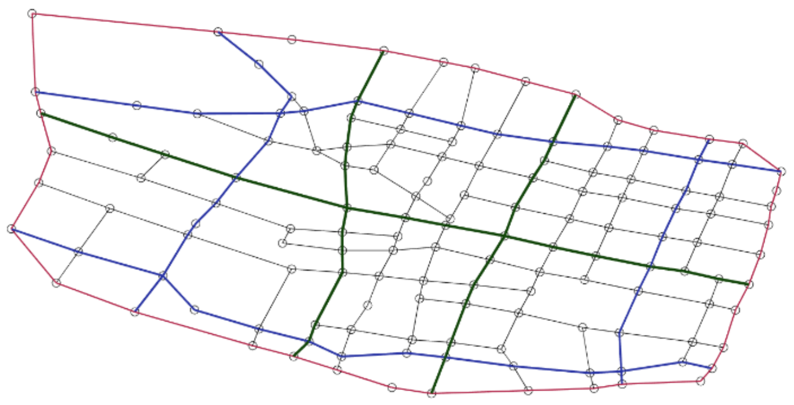
Figure 2-2 Road network diagram

2.2 Analysis of question one

Aiming at problem 1, that is, the optimal operation scheme for each sweeper to perform cleaning operation, the optimization goal is the shortest total mileage for all sweeper to complete all tasks. We need to process the distance data of each road, because considering that the width of each road is different, we add a certain width weight to the distance data of each road according to the width of the road. According to the optimization goal, we need to figure out the optimal path of each sweeper, and the optimization goal is to find the minimum total mileage.

2.3 Analysis of question two

According to question 2, in the case of a large amount of snow, there will be snow piling up points, then snowplows, trucks and sanitation workers for joint operations are needed, based on the problem of a snowplows optimal path, need to snow snowing accumulation point transport tasks, namely complete all snow accumulation point of optimal operation schemes of transportation, generally the optimization goal of snow shoveling snow and transportation task the shortest total mileage. According to the optimization goal, the optimal path of the snow plow and the optimal path of the transport vehicle (the transport vehicle and the sanitation worker work together) should be obtained, and the minimum value of the total driving distance of the two types of vehicles is the optimization goal.

2.4 Analysis of question three

For question three, in the case of parking spaces on the road, we divided the priority level according to the number of parking spaces on each road, cleaned the road according to the priority level, on the basis of question one, considering the constraints of priority division, we solved the shortest time required for the cleaning vehicle to complete the cleaning task.

2.5 Analysis of question four

According to question four, different sections have different priority levels. Based on question three, we carried out cleaning according to the priority level of the section, so as to first meet the highest level section and then meet the secondary section. According to the optimization goal and level constraint conditions, the shortest time required for the cleaning vehicle to complete the cleaning task is solved.

3. Symbol and Assumptions

3.1 Symbol Description

Symbol	Meaning
$v_1, v_2 \dots v_n$	Vertex
V	Vertex set
A	Arc set
K	Path set
k	Path
C	Arc length
D_{ij}	Edge right collection
E	Adjacency matrix for the edge weights
L	Distance
$\lvert A \rvert$	The number of directed arcs in arc set A
M	The set of cleaning vehicle
m	The MTH cleaning vehicle

K'	The arc set A is divided into different orders of priority
A^p	The set of arcs at P
T^p	Time for all vehicles to complete all services in A^p
N_p	The weights corresponding to T^p
t_0^m	Service start time of cleaning vehicle M
t_p^m	The time the cleaning vehicle M completes service at A^p
g_{ij}	Time required for cleaning vehicles to complete service in arc $(v_i，v_j)$
h_{ij}	The time required to clear a vehicle out of service in an arc $(v_i，v_j)$

3.2 Fundamental assumptions

1. Suppose the network given in the question is an operation area and a station, the location of which has been planned.
2. Suppose that all sweepers, plows, transporters, and sanitation workers operate at the same speed.
3. Suppose in the cleaning process, each path corresponds to a car, that is, each car only serves the same path. Vehicles start from the station, complete a path cleaning operation, and finally return to the station.
4. Suppose that the section is visited multiple times, but the section is served only once.
5. Suppose that all sections of the road are two-way lanes and there is a separation zone between them.

4. Model

Through the analysis of the problem, that is, the study of the best path for snow removal vehicles, we based on the graph theory analysis, study the best scheme of the shortest path. For road problems, we established the CARP [1-3] (Capacitated Arc Routing Problem) problem for analysis because of the assumption that the road is a two-way lane, that is, the road problem becomes a specific vehicle path problem - the arc path problem, and there are certain constraints. For the optimal method, ant colony, particle swarm, taboo search, genetic algorithm and simulated annealing algorithm can be used. Based on solving target and constraint conditions, we use Genetic algorithm and Dijkstra algorithm to obtain the optimal solution.

4.1 CARP questions and coordinates as distance

4.1.1 CARP questions

The CARP problem is based on the arc path problem, which can be defined as: given a connected network with a series of edges (or arcs) and points that require service, several vehicles take a particular point in the network as a field station, and the vehicle starts from the field station to serve the required side (or arc) in turn and eventually returns to the field station. The problem is to determine a reasonable set of driving paths that satisfy: all the required sides (or arcs) are served and each side (or arc) is served only once, and the total demand for services on the driving path does not exceed the capacity of the vehicle, so that the total cost of the whole process (usually the total mileage) is minimized.

4.1.2 Coordinate to distance

From the road network diagram obtained in section 2, it can be concluded that the entire snow cleaning process is based on the intersection vertex and the connection of the intersection. Before the establishment of the model, each intersection and its connection relationship in the road network diagram should be transformed into a weighted directed graph in mathematics, and the weight of each side is the distance between two intersections. Using Floyd algorithm[4] to convert coordinates into distance, plus the width weight of the lane, the distance weight of the two intersections is finally obtained.

Suppose that Vertex set $v = \{v_1, v_2 \dots v_n\}$is each intersection, the road between two adjacent intersections is the distance weight of the two intersections. We use D_{ij} to stand for weights, then the set of edge weights $D = \{d_{11}, d_{12}, d_{13} \dots d_{ij} \dots\}$. Drawing $G = (v, D)$, let the adjacency matrix of the weights be E.

$$E = \begin{pmatrix} e_{11} & \cdots & e_{1n} \\ \vdots & \ddots & \vdots \\ e_{n1} & \cdots & e_{nn} \end{pmatrix}$$

e$_{ij}$ is the distance weight of each side, that is $e_{ij} = d_{ij}$, and set up

$$d_{ij} = \begin{cases} \text{Weighting distance of } v_i, v_j & \text{(i, j adjacent access)} \\ 0 & \text{(i = j)} \\ \infty & \text{(i, j not adjacent or without acces)} \end{cases}$$

Where $v_i$and v_j are the two intersection points. Floyd algorithm is applied to solve the distance weight.

4.2 Model of question one

In the previous section, we have obtained the connection relation and distance weight of an intersection. According to the relevant knowledge of graph theory, we can transform the obtained road network graph into a directed graph, as is shown in Figure 4-1.

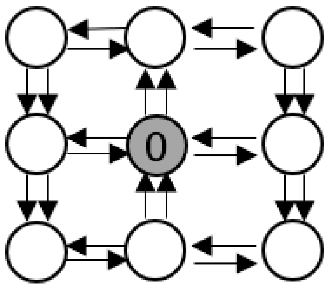

Figure 4-1 directed graph

If the directed arc (v_i, v_j) belongs to path k and is accessed from i to j, it is 1; otherwise, it is 0:

$$x_{ijk} = \begin{cases} 1, & \text{the arc } (v_i, v_j) \text{ belongs to path k and is accessed from i to j} \\ 0, & \text{otherwise} \end{cases}$$

Arc (v_i, v_j) belongs to path k and is 1 when served from i to j, otherwise 0:

$$y_{ijk} = \begin{cases} 1, & \text{the arc } (v_i, v_j) \text{ belongs to path k and is served from i to j} \\ 0, & \text{otherwise} \end{cases}$$

Measured by the total distance, the optimal solution is obtained by the objective function and constraint conditions.

The objective function：

$$\min\ L = \sum_{k\in K} \sum_{(v_i, v_j)\in A} C_{ij} x_{ijk} \tag{4-1}$$

The constraint s.t:

$$\sum_{\{v_j:(v_i,v_j)\in A\}} x_{ijk} = \sum_{\{v_j:(v_i,v_j)\in A\}} x_{jik} (v_i \in V, k \in K) \tag{4-2}$$

$$\sum_{k\in K} y_{ijk} = 1((v_i, v_j) \in A) \tag{4-3}$$

$$x_{ijk} \geq y_{ijk} ((v_i, v_j) \in A, k \in K) \tag{4-4}$$

$$\sum_{\{v_j:(v_0,v_j)\in A\}} x_{0jk} = 1(k \in K) \tag{4-5}$$

$$\sum_{\{v_i:(v_i,v_0)\in A\}} x_{i0k} = 1(k \in K) \tag{4-6}$$

$$\sum_{v_i, v_j \in S} x_{ijk} \leq |S| - 1 + |V|^2 u_k^s (S \subseteq V / \{v_0\}, S \neq \varnothing, k \in K) \tag{4-7}$$

$$\sum_{v_i \in S}\sum_{v_j \notin S} x_{ijk} \geq 1 - b_k^s (S \subseteq V/\{v_0\}, S \neq \varnothing, k \in K) \tag{4-8}$$

$$u_k^s + b_k^s \leq 1 (S \subseteq V/\{v_0\}, S \neq \varnothing, k \in K) \tag{4-9}$$

$$u_k^s, b_k^s \in \{0,1\} (S \subseteq V/\{v_0\}, S \neq \varnothing, k \in K) \tag{4-10}$$

$$x_{ijk}, y_{ijk} \in \{0,1\} ((v_{i,} v_j) \in A, k \in K) \tag{4-11}$$

The objective function (4-1) means to minimize the total distance traveled by the sweeper; Constraint formula (4-2) represents the conservation of node flow; Constraint formula (4-3) ensures that demand arcs are serviced; Constraint formula (4-4) indicates that the number of visits of each road section is not less than the number of services; Constraints (4-5) and (4-6) mean that all vehicles start from the station and finally return to the station;(4-7) to (4-10) represent loop elimination constraints, and (4-11) is a binary constraint.

4.3 Model of question two

According to the analysis of the problem, the second problem is that on the basis of the first problem, the transport vehicle needs to start from the station to transport to the snow accumulation point, and carry the snow back to the station when the transport vehicle is full.

On the basis of question 1, a new vertex set $W = \{w_1, w_2 \dots w_n\}$, where w_0 represents the station. Based on the data in section 4.1, the shortest path z_{i0} of each snow accumulation point leaving the station is calculated. Assuming that the capacity of the transporter is large, the number of service times for each snow accumulation point is 1. Then the model with constraints is:

The objective function：

$$\min\ L = \sum_{k \in K} \sum_{(v_{i,} v_j) \in A} C_{ij} x_{ijk} + \sum_{i=1}^{30} 2 z_{i0} \tag{4-12}$$

The constraint s.t.:

$$\sum_{\{v_j : (v_{i,} v_j) \in A\}} x_{ijk} = \sum_{\{v_j : (v_{i,} v_j) \in A\}} x_{jik} (v_i \in V, k \in K) \tag{4-13}$$

$$\sum_{k \in K} y_{ijk} = 1 ((v_{i,} v_j) \in A) \tag{4-14}$$

$$x_{ijk} \geq y_{ijk} ((v_{i,} v_j) \in A, k \in K) \tag{4-15}$$

$$\sum_{\{v_j : (v_{0,} v_j) \in A\}} x_{0jk} = 1 (k \in K) \tag{4-16}$$

$$\sum_{\{v_i:(v_i,v_0)\in A\}} x_{i0k} = 1(k\in K) \tag{4-17}$$

$$\sum_{v_i,v_j\in S} x_{ijk} \le |S|-1+|V|^2 u_k^s (S\subseteq V/\{v_0\}, S\ne\varnothing, k\in K) \tag{4-18}$$

$$\sum_{v_i\in S}\sum_{v_j\notin S} x_{ijk} \ge 1-b_k^s (S\subseteq V/\{v_0\}, S\ne\varnothing, k\in K) \tag{4-19}$$

$$u_k^s + b_k^s \le 1(S\subseteq V/\{v_0\}, S\ne\varnothing, k\in K) \tag{4-20}$$

$$u_k^s, b_k^s \in \{0,1\}(S\subseteq V/\{v_0\}, S\ne\varnothing, k\in K) \tag{4-21}$$

$$x_{ijk}, y_{ijk} \in \{0,1\}((v_i, v_j)\in A, k\in K) \tag{4-22}$$

Equation (4-12) represents the minimum total distance traveled by snowplow and transporter.

4.4 Model of question three and four

It can be known from the problem analysis, questions 3 and 4 are based on question one, including the priority of the demand arc (the priority of the road). Set arcs A divided into K' arc subsets by service level, $A^1 \cup A^2 \cup \ldots \cup A^{K'} = A$, and $A^i \cap A^j = \varnothing$, among them $i \neq j$. For each sweeper, the order in which all arcs in the arc set A^i are served has priority A^{i+1}. In addition, the snow removal vehicle needs to return to the station after performing the cleaning task, Set $A^{K'+1}$ as the set of directed arcs contained in the shortest path from all nodes in $A^{K'}$ to v_0. For each arc set A^p, $p = 1, \ldots, K' + 1$, let T^p be the cleaning completion time of the arc set A^p. In order to make the model have path continuity under the priority limit. Introduce the virtual point v_x on the basis of the figure, so that the cleaning vehicle can complete the service of the arc A^p, back to v_x, service A^{p+1}again, as the figure shows, where v_x is connected to any other vertices, its directed arc is $A_1 = \{(v_x, v_i): v_i \in V\}$, $A_2 = \{(v_i, v_x): v_i \in V\}$. Introduce new parameters based on the model of problem one, And define y_{ij}^{pm} as the number of empty-driving visits of arc (v_i, v_j) in arc set $A\cup A_1\cup A_2$ when cleaning vehicle m performs cleaning task on A^p; x_{ij}^{pm} —When cleaning vehicle m performs cleaning service on A^p middle arc (v_i, v_j), then 1, otherwise 0. w_{ij}^m is the total number of times that the sweeping vehicle passes through arc (v_i, v_j) in arc set $A\cup A_1\cup A_2$. According to the optimization goal, the model is established as:

The objective function:

$$\min \sum_{p=1}^{K'+1} N_p T^p \tag{4-23}$$

The constraint s.t.:

$$T^p \geq t_p^m (p=1,...,K'+1, m \in M) \tag{4-24}$$

$$t_p^m = t_{p-1}^m + \sum_{(v_i,v_j)\in A} (g_{ij} x_{ij}^{pm} + h_{ij} y_{ij}^{pm})(p=1,...,K'+1, m \in M) \tag{4-25}$$

$$t_0^m = 0(m \in M) \tag{4-26}$$

$$\sum_{p=1}^{K'} \sum_{m \in M} x_{ij}^{pm} = 1((v_{i,} v_j) \in A) \tag{4-27}$$

$$\sum_{v_j \in V \cup \{v_x\}} (x_{ij}^{pm} + y_{ij}^{pm}) = \sum_{v_j \in V \cup \{v_x\}} (x_{ji}^{pm} + y_{ji}^{pm})(v_i \in V \cup \{v_x\}, p=1,...,K'+1, m \in M) \tag{4-28}$$

$$\sum_{(v_i,v_j)\in A \cup A_1 \cup A_2} w_{ij}^m = \sum_{(v_i,v_j)\in A \cup A_1 \cup A_2} w_{ji}^m (v_i \in V \cup \{v_x\}, m \in M) \tag{4-29}$$

$$x_{ij}^{pm} + y_{ij}^{pm} \leq w_{ij}^m \leq |A| (x_{ij}^{pm} + y_{ij}^{pm})((v_{i,} v_j) \in A \cup A_1, p=1,...,K'+1, m \in M) \tag{4-30}$$

$$x_{ij}^{pm} + y_{ij}^{pm} \leq w_{ix}^m ((v_{i,} v_j) \in A, p=1,...,K'+1, m \in M) \tag{4-31}$$

$$\sum_{v_i \in V} y_{xi}^{pm} = 1(p=1,...,K'+1, m \in M) \tag{4-32}$$

$$\sum_{v_i \in V} y_{ix}^{pm} = 1(p=1,...,K'+1, m \in M) \tag{4-33}$$

$$y_{xi}^{pm} = y_{ix}^{pm} (v_i \in V, p=1,...,K'+1, m \in M) \tag{4-34}$$

$$y_{x0}^{1m} = 1(m \in M) \tag{4-35}$$

$$y_{0x}^{K'+1m} = 1(m \in M) \tag{4-36}$$

$$x_{ij}^{pm} \in \{0,1\}((v_{i,} v_j) \in A, p=1,...,K'+1, m \in M) \tag{4-37}$$

$$y_{ij}^{pm} \geq 0 \text{ and as an integer}((v_{i,} v_j) \in A \cup A_1 \cup A_2, p=1,...,K'+1, m \in M) \tag{4-38}$$

$$T^p \geq 0(p=1,...,K'+1) \tag{4-39}$$

$$t_p^m \geq 0(p=1,...,K'+1) \tag{4-40}$$

The objective function in Equation (4-23) represents the minimum value of the time to complete all the road cleaning tasks. Constraint (4-24) is that the cleaning time of a certain service level should not be less than that of any cleaning vehicle in the road network of this level. Constraints (4-25) and (4-26) define the time at which a cleaning vehicle starts and finishes cleaning at each service level. The constraint (4-27) indicates that a directed arc can only be served by one sweeper. Constraint (4-28) is the continuity constraint of the cleaning vehicle's driving path. The constraint (4-29) is node flow conservation. Constraints (4-30) are non-negative constraints on the flow of the cleaning vehicle on any arc. Constraint (4-31) is to cancel the constraint of the closed sub-loop. Equations (4-32) and (4-33) indicate that the sub-path of any cleaning vehicle in each service level begins and ends at the virtual node v_x. Constraint (4-34) indicates that the cleaning vehicle will drive back to the node to perform the next level of cleaning after completing each level of cleaning task at a node and driving to the virtual point on the original road. Constraints (4-35) and (4-36) mean that the cleaning vehicle starts from the station to complete all levels of cleaning tasks and returns to the station. Equations (4-37) to (4-40) represent the value constraints of each variable. However, because there are a large number of variables and complex constraints in the model, the calculation is more complex. Therefore, parallel mode computation is adopted for optimization. According to the above model, the service level A^p is calculated respectively. The model for arc set A^p is:

The objective function：

$$\min \quad T^p \tag{4-41}$$

The constraint s.t :

$$T^p \ge t_p^m (m \in M) \tag{4-42}$$

$$t_p^m = S_p^m + \sum_{(v_i, v_j) \in A} (g_{ij} x_{ij}^{pm} + h_{ij} y_{ij}^{pm})(m \in M) \tag{4-43}$$

$$\sum_{m \in M} x_{ij}^{pm} = 1((v_{i,} v_j) \in A) \tag{4-44}$$

$$\sum_{v_j \in V \cup \{v_x\}} (x_{ij}^{pm} + y_{ij}^{pm}) = \sum_{v_j \in V \cup \{v_x\}} (x_{ji}^{pm} + y_{ji}^{pm})(v_i \in V \cup \{v_x\}, m \in M) \tag{4-45}$$

$$\sum_{(v_{i,} v_j) \in A \cup A_1 \cup A_2} w_{ij}^m = \sum_{(v_{i,} v_j) \in A \cup A_1 \cup A_2} w_{ji}^m (v_i \in V \cup \{v_x\}, m \in M) \tag{4-46}$$

$$x_{ij}^{pm} + y_{ij}^{pm} \le w_{ij}^m \le |A| (x_{ij}^{pm} + y_{ij}^{pm})((v_{i,} v_j) \in A \cup A_1, m \in M) \tag{4-47}$$

$$x_{ij}^{pm} + y_{ij}^{pm} \le w_{ix}^m ((v_{i,} v_j) \in A, m \in M) \tag{4-48}$$

$$\sum_{v_i \in V} y_{xi}^{pm} = 1(m \in M) \tag{4-49}$$

$$\sum_{v_i \in V} y_{ix}^{pm} = 1(m \in M) \tag{4-50}$$

$$y_{xStart_p^m}^{pm} = 1(m \in M) \tag{4-51}$$

$$x_{ij}^{pm} \in \{0,1\}((v_{i,} v_j) \in A, m \in M) \tag{4-52}$$

$$y_{ij}^{pm} \ge 0 \text{ and as an integer}((v_{i,} v_j) \in A \cup A_1 \cup A_2, m \in M) \tag{4-53}$$

$$T^p \ge 0 \tag{4-54}$$

$$t_p^m \ge 0 \tag{4-55}$$

In this model, the objective function is the minimum value of the time it takes for the cleaning vehicle to complete snow removal at the level A^P. S_p^m is the time when the cleaning vehicle m starts snow removal at A^P.

5. Test the Models

5.1 The Dijkstra algorithm of minimum path solution

The algorithms to find the shortest path such as Floy dalgorithm and Dijkstra algorithm [5], are very sophisticated algorithms. In this question we will take Dijkstra algorithm. The basic idea is as follows:

Construct a weighted graph $G = (V, E, M)$, among them, Vertex set $V = \{v_1, \cdots, v_n\}$, $v_1, \cdots, v_n$ means each location; E is the set of edges;Adjacency matrix $W = (w_{ij})_{n\times n}$, w_{ij} represents the distance of the path between the vertex u_i and v_j, If there is no path between vertices v_i and v_j, $w_{ij} = \infty$。 The problem is to find the path with the smallest weight between the two vertices u_0 and v_0 in the weighted graph G. This road is called the shortest distance between u_0 and v_0, The idea of using Dijkstra's algorithm is to order from near far from u_0,Find the shortest path and distance of each vertex from $u_0$to G in turn, until v_0 (or until all vertices of G),algorithm end. In order to avoid repetition and retain the calculation information of each step, a labeling algorithm is adopted. Here is the algorithm.

(1)Set $l(u_0 = 0)$, $v \neq u_0$, set $l(v)$=∞,$S_0 = \{u_0\}$, $i = 0$.

(2) Arbitrary $v \in \overline{s_i}(\overline{s_i} = V/S_i)$, use

$$\min_{u\in s_i}\{l(v), l(u) + w(uv)\}$$

instead $l(v)$, Here w_{uv} represents the weight between vertices u and v. Calculation $\min_{u\in s_i}\{l(v)\}$, Let the vertex that reaches this minimum value be u_{i+1}, set $S_{i+1} = S_i \cup \{u_{i+1}\}$

(3)If $i = |V| - 1$, Then stop; if $i < |V| - 1$, Use $i+1$ instead of i, turn to (2).

At the end of the algorithm, The distance from u_0 to each vertex v is given by the last successive label $l(v)$ of v. The label $l(v)$ before v enters S_i is called the T label, When v enters S_i, the label $l(v)$ is called P label。 The algorithm is to continuously modify the T label of each vertex until the P label is obtained。 If the algorithm is running, the edge from which each vertex gets the P label is marked on the graph. At the end of the algorithm, the shortest path from u_0 to each vertex is also marked on the graph.

5.2 Introduction to Genetic algorithm

Genetic Algorithms [6,7], GA is a search algorithm based on the principle of natural selection and natural genetic mechanism. It simulates the life evolution mechanism in nature, and optimizes to achieve specific goals in artificial systems. The essence of genetic algorithm is to use group search technology to evolve from generation to generation according to the principle of survival of the fittest, and finally obtain the optimal or quasi-optimal solution。 It must do the following: The generation of the initial group, the fitness of each individual, and the selection of the best individual according to the principle of survival of the fittest、the selected good individuals are paired in pairs, and the next generation population is generated by randomly crossing the genes of their chromosomes and randomly mutating the genes of chromosomes, according to this method, the population evolves from generation to generation until the evolution termination condition is met. The implementation method is as follows:

(1) Determine the feasible solution domain according to the specific problem, and determine a coding method that can represent each solution of the feasible solution domain with a numerical string.

(2) There should be a basis for measuring the quality of each solution, which is expressed by a function, called the fitness function, which is generally composed of an objective function.

(3) Determine the evolution parameter population size M, crossover probability p_c, mutation probability p_m, evolution termination condition.

In order to facilitate the solution, generally the number of individuals in each generation of the group is equal. The larger the group size, the easier it is to find the optimal solution, but due to the limitation of the computing power of the computer, the larger the group size, the time required for calculation will increase accordingly. The evolution termination condition refers to when the evolution ends, it can set a certain generation of evolution to end, or it can be determined by finding out whether the approximate optimal solution meets the accuracy requirements.

5.3 Solution to problem number one

Since the study of the best path for cleaning vehicles is a typical CARP problem, the general solutions to this problem can be divided into precise algorithms, heuristic algorithms and sub-heuristic algorithms. Sub-heuristic algorithms mainly include genetic algorithm, tabu search, ant colony algorithm, simulated annealing, etc. to compare its advantages and disadvantages, we choose genetic algorithm to solve.

5.3.1 Algorithm coding design

Generally, in the coding design of genetic algorithm, a chromosome represents a solution, but when designing a genetic algorithm to solve problem 1, a chromosome in the algorithm does not represent a solution of the problem, but only represents a part of the solution, that is, a sweeper Service route, and all chromosomes in the population collectively represent a complete solution to the problem. For the coding method of chromosomes, the width of a certain road can be converted into the service demand of the route with a certain weight, and then the demand of all routes can be obtained, and each route is numbered to obtain the demand arc number, so a series of chromosomes The integer series code of, which represents the number of the demand arc served on the path of a certain snow removal vehicle. The coding process of chromosomes is shown in Figure 5-1:

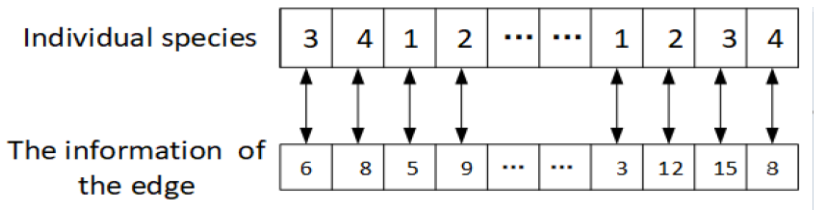

Figure 5-1 Chromosome coding map

As shown in Figure 5-1, The individual length of the population represents the number of snow removal vehicles on the path，If the i-th place is 1, it means that the demand for the road section is 1, that is, the number of times the vehicle service is needed is 1, and the generated chromosome $p(k)=\{6,8,5,9\cdots 3,12,15,8\}$ .The chromosome indicates that the code of the vehicle serving in a certain driving path is $6,8,5,9\cdots 3,12,15,8$, and suppose that the start and end of the vehicle are both stations.

5.3.2 Construct fitness function

For question one, the path distance traveled by the vehicle is taken as the objective function and at the same time as the fitness function. The calculation formula of the fitness value is as follows：

$$F(k)=1/\sum_{i=1}^{n} dist(d_i)$$

In the above formula, $d_i$represents the shortest path required to serve each demand arc. Therefore, the greater the fitness value, the greater the probability that the chromosomes will survive evolution. Calculation the fitness value is to calculate the total travel path of the vehicle, which can be calculated by the algorithm mentioned above. The calculation method is as follows:

For a chromosome $p(k)=\{5,2,7,9\}$， The driving route rule of the vehicle is: the vehicle departs from the station, and the station is recorded as 0， For the first time, the demand arc (v_5,v_6) needs to be serviced, First compare the sizes of d_{50} and d_{60}, if the $d_{50} < d_{60}$ vehicle chooses to serve the demand arc from point 2 to (v_5,v_6)， After serving the demand arc, record the shortest path and shortest distance it travels, and then serve the next demand arc, At this time, the second demand arc is (v_2,v_3)， Compare the size of d_{25}andd_{35},if $d_{25} > d_{35}$, Then choose to drive into the demand arc from node 3 and serve it until all demand arcs are serviced. The algorithm process is shown in Figure 5-2:

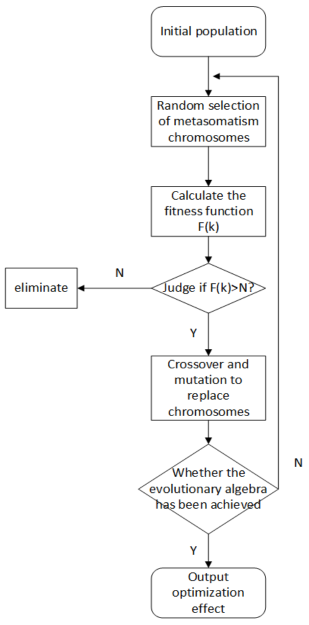
Initial population
Random selection of metasomatism chromosomes
Calculate the fitness function F(k)
eliminate
N
Judge if F(k)>N?
Y
Crossover and mutation to replace chromosomes
N
Whether the evolutionary algebra has been achieved
Y
Output optimization effect

Figure 5-2 Genetic algorithm flowchart

5.3.3 Solution result analysis

Program the algorithm with MATLAB according to the previous algorithm。The main parameters of the experiment：The size of the population G is 25， The crossover probability range during population initialization is set to 0.8, and the mutation probability range is set to 0.002. The algorithm iteration diagram of the experiment is shown in Figure 5-3：

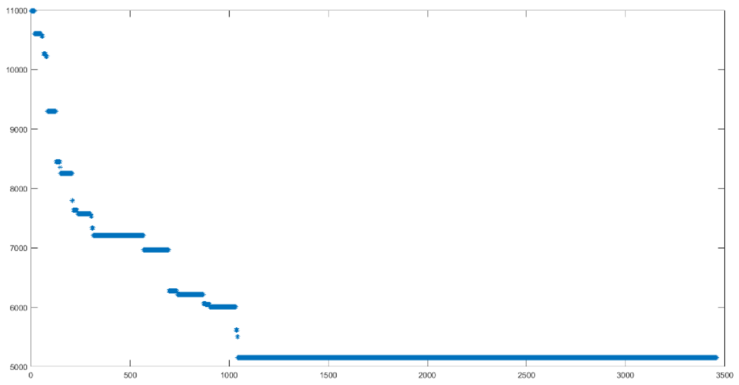

Figure 5-3 The relationship between the optimal objective function value and the number of iterations

Solution to question One，When writing genetic algorithms，We selected 25 populations, each containing 10 chromosomes, that is, the driving route of 10 snow removal vehicles，Through continuous iteration, the optimal solution is found. Since there are too many paths and each path is repeated, two paths are used to illustrate, as shown in Figure 5-4：

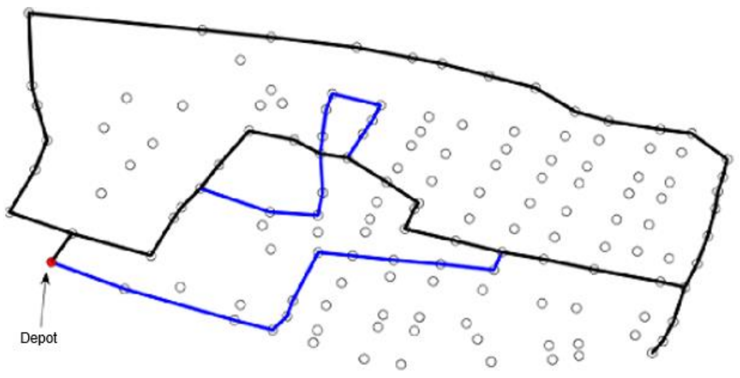

Figure 5-4 Road map of sweeper

The specific routes and distances of the two paths are shown in Table 5-1：

Table 5-1 The specific routes and distances of the two paths

Vehicle number	Path of the vehicle	Total distance traveled by vehicle (km)
1	{1→2→6→7→128→129 →126→108→109→119 →118→117→116→113 →112→111→110→103 →93→92→91→90→77 →40→77→32→31→26 →21→16→15→14→13 →11→8→1}	28.69842
2	{1→2→3→4→131→134 →135→139→138→140→115→114 →98→97→84→83→71→70 →61→60→59→58→57→53 →52→51→50→49→48→49 →50→51→47→41→40→77 →90→91→92→93→103→110 →119→120→124→125→129→128 →7→6→2→1}	44.1678

As shown in the figure, after iterative optimization, the path of each vehicle starts from the station, passes through the optimal path to serve all demand paths, and then returns to the station with the shortest path. it can be seen that there are overlaps between different road maps. This is because different road sections require different service times for snow removal vehicles. For those paths that are in high demand, the more the paths are repeated， Under the iterative optimization of genetic algorithm, the total distance of all snow removal vehicles is minimized under the condition that the demand of all paths is satisfied, and the minimum total distance is:

$$L_{min}=2178.1347 \text{ km}$$

5.4 Solution to problem number two

For problem two, it is solved on the basis of problem one. In the case of heavy snow, there will be snow accumulation points. At this time, snow shovel, transport vehicle and sanitation workers are required for joint operation. Based on the optimal path of the snow shovel in problem one, it is necessary to carry out the snow transportation task for the snow accumulation points, that is, the optimal operation plan for the transportation of all snow accumulation points. The minimum value of the total driving distance of the two types of vehicles is the optimization goal. Since the first question has obtained the shortest path of all snowplows, it is only necessary to minimize the total path of the transport vehicle.

On the basis of the first question model algorithm, plus the constraint conditions of the second question model, the optimal route map for the transport vehicle to start from the station and then return to the station is obtained. As shown in Figure 5-5. The points circled in blue in the figure are snow accumulation points. The detailed table of the three paths is shown in Table 5-2.

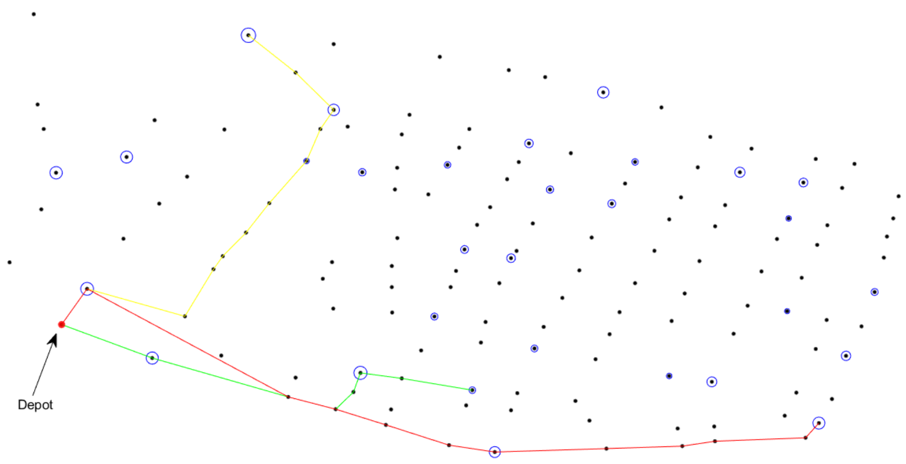

Figure 5-5Three optimal roadmaps

Table 5-2 The detailed table of the three paths

Vehicle number	Path of the vehicle	Total distance traveled by vehicle (km)
1	{1→8→11→13→14→15 →19→24}	7.5564
2	{1→2→6→7→128→129 →125→124→123→122→141→138}	10.5465
3	{1→2→11→13→17→22 →27→36→37→44→48→49}	12.7624

Based on the above analysis, the path traveled by each transport vehicle not only contains one potential snow point, and there will be different repetitions of the transport vehicle path for the snow point where there is a lot of snow. Therefore, it can be judged that under the iteration of the genetic algorithm, a set of better travel paths of the transport vehicles can be obtained, and the total distance of the optimal path of the transport vehicles obtained is：

$$\sum_{i=1}^{30} 2z_{i0} = 597.8689 \text{ km}$$

The total distance of the optimal path for all vehicles is:

$$L_{min} = 2746.0036 \text{ km}$$

5.5 Solution to problem number three

For question three, according to the priority of each road section, a road network diagram with priority is generated, as shown in Figure 5-6:

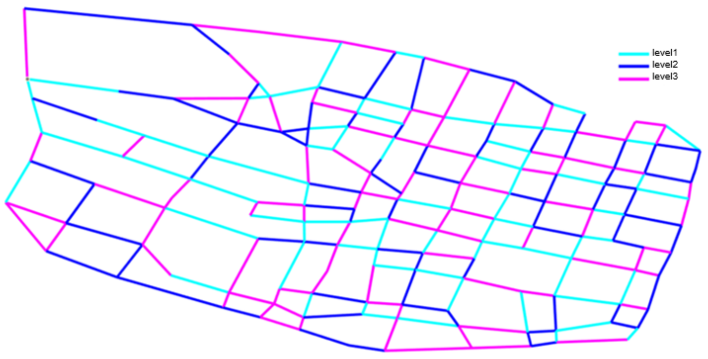

Figure 5-6 Three-level road network diagram

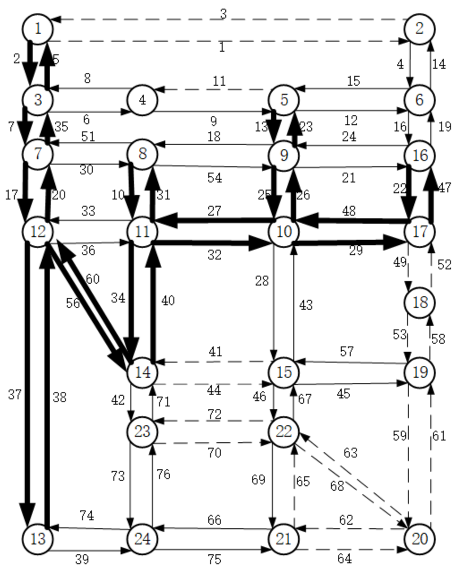

Figure 5-7 Part of the three-level road network conversion directed graph

Assume that the station has 10 sweeping vehicles, and the service speed and empty speed of the vehicles are both 40km/h. According to the graph conversion method, the road network graph is converted into a directed graph. Because of the complexity of the data, we select a part of the directed graph for illustration, as shown in Figure 5-7. The black solid line and bold arrow represent the first priority level. The solid arrow represents the second priority level, and the black dashed arrow represents the third priority level.

5.5.1 New heuristic algorithm based on parallel and genetic algorithm

According to the idea of parallel computing, it is necessary to arrange all cleaning vehicles to service the arcs of the first service level in the road network first, and then perform cleaning services for the arcs of the lower service level. The solution of the model can be regarded as a continuous solution to multiple arcs. The minimum value of the cleaning completion time is the target m-RPP problem [8].

The path optimization process in which multiple sweepers perform cleaning tasks on each level of sub-road network is called a sub-process, that is, the parallel computing method needs to execute several sub-processes continuously. In the first sub-process, all vehicles start at the station node. In addition, the starting node of all vehicles at the beginning of other sub-processes is uncertain (the starting node position and number are uncertain), and each vehicle is cleaned. The number of arcs that the car may serve is also uncertain. The continuous optimization of several sub-processes, combined with the consideration of the sub-process's optimization goal—the minimum cleaning completion time, failed to find the existing suitable heuristic algorithm to solve the problem. In response to this problem, this section constructs an effective new heuristic algorithm based on the concept of chromosome coding in genetic algorithm and the idea of genetic and crossover to optimize the path of parallel computing methods. The algorithm process is as follows：

1 Algorithm coding

Take the cleaning vehicle performing the first-level sub-road network cleaning service as an example. There are a total of 50 directed arcs in the sub-road network at this level. They are sorted according to the arc number in the road network, and the natural number 1-50 represents the sequence number of each arc. Figure 5-8 shows the coding method of snow removal service path for a certain sweeper in this level:

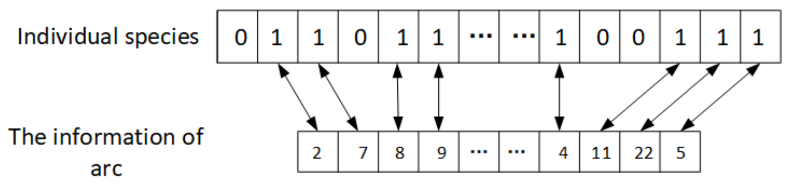

Figure 5-8 Chromosome coding map

The chromosome uses binary coding, and the length of the chromosome is equal to the number of first-level directed arcs in the road network. Therefore, the chromosome has a total of 50 genes. The position of the gene in the chromosome from right to left corresponds to the number of the directed arc one by one. The value of 0 and 1 of the gene in the chromosome indicates whether the first-level directed arc is served by the sweeper. When the value is 1, it means that it is served.

2 Initialization and gene exchange

Initialize the 10 chromosomes corresponding to 10 sweepers to ensure that each arc is served by only one snow removal vehicle. A set of chromosomes represents a solution to the problem. The number of first-level directed arcs is 50, so the length of the coloring question corresponding to each snow removal vehicle is 50, the second-level and third-level directed arcs are 80 and 111 respectively, and the lengths of chromosomes are respectively 80 and 111. Initialize. The rules are the same.

Each chromosome is marked with λ_i for the number of directed arcs served. t_i represents the shortest service time for the snow removal vehicle to complete these directional arc snow removal tasks. Taking the calculation of t_1 as an example, the snow removal vehicle starts from the stop, that is $Start_p^m = v_0$. Use the Dijkstra method to find the shortest path, search for the starting node of all directed arcs with gene value 1 in chromosome c1, and find the one with the smallest empty time to node v_0 and get the corresponding shortest path, save this node and path information. Then the connected node of this node is $Start_p^m$, repeat the above process until all the starting and ending nodes of the directed arc that need to be served are searched, and the shortest service time t_1 of chromosome c_1 is obtained (including the service time and idle time on the directed arc).

Compare the shortest service time of 10 chromosomes, get the longest chromosome c_{max} and the shortest chromosome c_{min}, and perform gene exchange between the two. The exchange method uses a single point exchange on the same position of two chromosomes. A new set of chromosomes is obtained and the calculation of the shortest service time of the set of chromosomes and exchange operations are repeated until the number of iterations is met and the calculation is stopped. The exchange iterative process makes the maximum service time of the chromosome gradually converge to a minimum, and this optimization process satisfies that all arcs are served and only one snow removal vehicle is served.

3 Seeking object and algorithm flow chart

There are two situations for the optimization object of the parallel computing method. In the first case, the T^p of multiple snow removal vehicles in the sub-road network of each service level is independently optimized, and the iterative optimization of the higher-level sub-road network is completed, and the output is The node information and service completion time of snow removal vehicles are used as the input of iterative optimization in the next-level sub-road network. The calculation is performed from high to low. There is only one output and input correlation in the upper and lower independent optimization process. Finally, the integration of each sub-road network The service path of the snow removal vehicle is taken as the final path optimization result; in the second case, the chromosome group of each level of sub-road network completes a gene exchange, and the output data is used as the input of the next level of sub-road network path optimization, taking the service path of the entire road network As an optimal object. Therefore, the first case is called sub-process optimization, and the second is called full-process optimization. Figure 5-9 shows the difference between these two situations more vividly. This article adopts the whole process of optimization calculation.

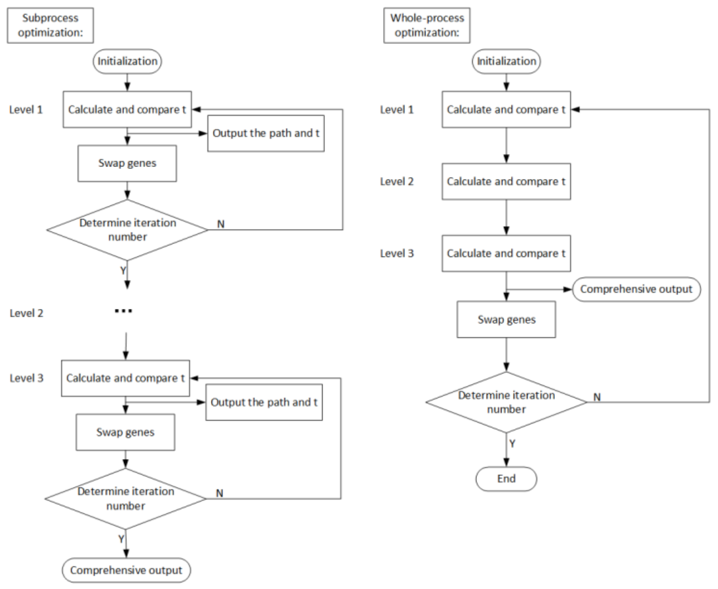

Figure 5-8 Flow chart of sub-process optimization and whole process optimization

5.2.2 Solution result analysis

Table 5-3 the path information obtained from the optimization calculation

Serial computing genetic algorithm to find the optimal path planning results				
Snow removal vehicles	Service path		Service completion time t_4^m (h)	Empty time t_d^m (h)
Vehicle 1	Level 1	1-2-11-13-14-15	5.82	0.9
	Level 2	113-112-111-118-120-124-125-129-128		
Vehicle 2	Level 1	133-134-131-4-2-1	6.08	0.85
	Level 2	7-128-129-108-105-104-92-93-90		
	Level 3	69-72-82-85-96-99-100-101-111-118-120-124-125		
Vehicle 3	Level 1	2-6-7-12-16	6.56	0.43
	Level 2	12-16-21-26-31-32-77-40		
	Level 3	124-120-118-111-101-100-99-96-85-82		
		⋮		

Table 5-3 is the path information obtained from the optimization calculation of the whole process of parallel calculation, which includes: the service path of all snow removal vehicles, as well as the time t_4^m for each vehicle to complete the snow removal and return to the station and the path empty time t_d^m.

According to the service path of the cleaning vehicle obtained from the solution process in Table 5-3, calculate the cleaning completion time of each cleaning vehicle in the sub-road network at all levels, and get Table 5-4. In the table, represents the time for the m-th snow removal vehicle to complete the snow removal service in the p-th level sub-road network, and T^p represents the time to complete the snow removal of the p-th level sub-road network.

Table 5-4 the cleaning completion time of each cleaning vehicle in the sub-road network

Whole-process optimization		
Service level	t(h)	T^p(h)
Level 1	3.05	4.09
	4.09	
	4.04	
	3.24	
	…	
Level 2	6.08	6.54
	6.48	
	6.32	
	6.54	
	…	
Level 3	9.12	9.12
	8.44	
	8.54	
	8.78	
	…	

Based on the above analysis, under the constraints of the three priority levels of service, the snow removal vehicle will take 6.99 hours to complete all the road networks.

5.6 Solution to problem number four

The same as the solution method of problem three, we will generate a road network diagram with 6 priority levels, as shown in Figure 5-9:

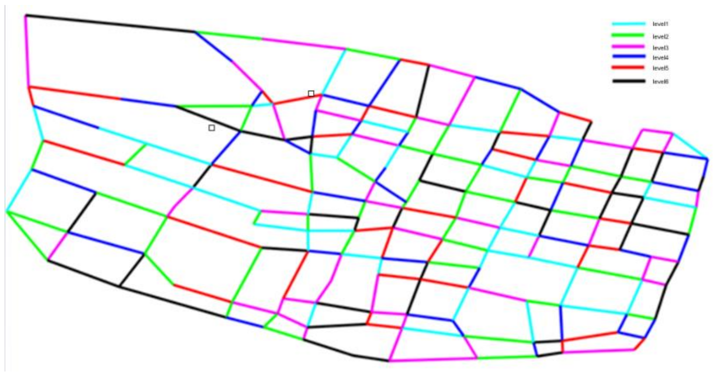

Figure 5-9 six-level road network diagram

Using the same algorithm as in problem three, the path information obtained from the optimization calculation in the whole process and the cleaning completion time of each cleaning vehicle in the sub-road network at all levels are shown in Table 5-5 and Table 5-6.

Table 5-5 the path information obtained from the optimization calculation

Serial computing genetic algorithm to find the optimal path planning results				
Snow removal vehicles	Service path		Service completion time (h)	Empty time (h)
Vehicle 1	Level 1	1-2-6-7	2.88	0.12
	Level 2	1-8-11-13-14-15-19-24		
Vehicle 2	Level 1	7-128-129-125-124-120-118-111-101-100	4.82	0.84
	Level 2	11-13-17-22-27-36-37-44-48-49		
	Level 3	6-7-128-129-125-124-120-118-111-101-100-99		
Vehicle 3	Level 1	14-18-23-28-35-38	5.14	0. 78
	Level 2	13-17-22-27-36-37-44-48-49		
	Level 3	128-129-125-124-120-118-111-101-100-99-96-85-82-72		
		⋮		

Table 5-6 the cleaning completion time of each cleaning vehicle in the sub-road network

Whole-process optimization		
Service level	t(h)	T_{max}(h)
Level 1	3.52	3.52
	3.52	
	3.44	
	3.24	
	…	
Level 2	4.18	4.46
	4.24	
	4.46	
	4.12	
	…	
Level 3	6.30	6.68
	6.24	
	6.68	
	6.47	
	…	
…….		

Based on the above analysis, under the constraints of the three priority levels of service, the snow removal vehicle will take 5.92 hours to complete all the road networks.

6.Strengths and Weakness

6.1 Strengths

1. The model in this paper focuses on data processing, presented in the form of tables and graphs, which greatly improves query efficiency.
2. The paper gives a large number of tables and diagrams, which are used to express the path and path planning, which are intuitive and easy to understand, and the results are accurate.
3. Models and algorithms are highly operable and easy to promote and apply. The constraint conditions can be changed according to specific requirements, and the algorithm can be changed for calculation.
4. Applying genetic algorithms and new heuristic algorithms based on genetic and parallel algorithms have short running time and accurate results.

6.2 Weakness

1. In the process of modeling and programming, the data used is an approximation of the actual data, so the results obtained may have a certain gap with the actual situation.
2. The amount of calculation is large, and the combination of paths is prone to exponential growth of the combination path.
3. The topic is a solution under ideal conditions. In real life, there are many other factors. The model can also take into account many variable factors. Since the question does not provide the corresponding reference factor materials, it is not considered in solving the problem.

7.Conclusion

The cleaning of urban snow roads is closely related to the normal operation of the city. How to deal with the snow quickly and effectively is what people pursue and hope. Under the different constraints of this question, the minimum total mileage and minimum time of vehicle operation are as follows, because there are too many specific programs for operation, they are not listed here:

	question1	question2
The shortest total distance /km	2178.1347	2746.0036
	question3	question4
The shortest time /h	6.99	5.92

It can be seen from the table that as the constraint conditions increase, the shortest total mileage or the shortest time of the corresponding operation will increase accordingly. The most important thing is the number of priority levels.

References

[1] Lacomme P , Prins C , Ramdane-Cherif W . Competitive Memetic Algorithms for Arc Routing Problems[J]. Annals of Operations Research, 2004, 131(1):159-185.

[2] P. Fernández de Córdoba, Raffi L M G , Sanchis J M . A heuristic algorithm based on Monte Carlo methods for the Rural Postman Problem[J]. Computers & Operations Research, 1998, 25(12):1097-1106.

[3] Kandula P , Wright J R . Designing Network Partitions to Improve Maintenance Routing[J]. Journal of Infrastructure Systems, 1997, 3(4):160-168.

[4] Wang J Y , Wright J R . Interactive Design of Service Routes[J]. Journal of Transportation Engineering, 1994, 120(6):897-913.

[5] Jenelius E , Petersen T , Mattsson L G . Importance and exposure in road network vulnerability analysis[J]. Transportation Research Part A Policy & Practice, 2006, 40(7):537-560.

[6] Hertz A , Laporte G , Mittaz M . A Tabu Search Heuristic for the Capacitated Arc Routing Problem[M]. INFORMS, 2000.

[7] Wen, Lea, Pearn. Approximate solutions for the capacitated arc routing problem[J]. Computers & Operations Research, 1989.

[8] Wang J Y , Wright J R . Interactive Design of Service Routes[J]. Journal of Transportation Engineering, 1994, 120(6):897-913.

Appendix

Question 1

```
clc,clear
link=xlsread("links.xls");
sj0=xlsread("Attachment 1.xlsx");
loc_nodes=xlsread("Attachment 1.xlsx");
startnode=link(:,1);
endnode=link(:,2);
width=link(:,3);
HU=zeros(241,3);
HU(:,1)=startnode;
HU(:,2)=endnode;
HU(:,3)=width/2;
ZOHU=sum(HU);
ZOHU1=ZOHU(1,3);
link1=zeros(141);
for n=1:241
    link1(startnode(n),endnode(n))=1;
end
link1=link1+link1';

x=sj0(:,2); x=x(:);
y=sj0(:,3); y=y(:);
kb=5;
sj=[x y];

sj=sj*pi/180;
d=zeros(141);
d1=zeros(141);
for i=1:140
    for j=i+1:141
        d(i,j)=6370*acos(cos(sj(i,1)-
sj(j,1))*cos(sj(i,2))*cos(sj(j,2))+sin(sj(i,2))*sin(s
j(j,2)));
    end
end
d=d+d';

for i=1:140
    for j=i+1:141
        if link1(i,j)==1
            d1(i,j)=d(i,j);
        end
        if link1(i,j)==0
            d1(i,j)=inf;
        end
    end
end
d1=d1+d1';




M_PI=3.14159265358979323846;

xlab=loc_nodes(:,2);
ylab=loc_nodes(:,3);
position=[xlab ylab];
L = 6381372 * M_PI * 2;
W = L;
H = L / 2;
mill = 2.3;
n=size(position,1);
new_position=[];
figure
for i =1:n
    lon=position(i,1);
    lat=position(i,2);
    x = lon * M_PI / 180;
    y = lat * M_PI / 180;
    y1 = 1.25 * log(tan(0.25 * M_PI + 0.4 * y));

    dikaerX(i) = (W / 2) + (W / (2 * M_PI)) * x ;
    dikaerY(i) = (H / 2) - (H / (2 * mill)) * y1 ;
    new_position(i,1)=dikaerX(i);
    new_position(i,2)=dikaerY(i);

plot(dikaerX(i),dikaerY(i),'o','color','k','Markersiz
e',8)
    hold on
end

path_0=[0];
path_z=[0];
qi=1;
path_0=[qi];
dist_0=zeros(L_1,1);
dist_z=[];
```

```
dist1=[];
dist2=[];
xx=[];
yy=[];

c_z=randperm(241);
k=5;
c=c_z(1:k);
c1=c_z(k:241);
for i=1:k
    [dist1,~]=mydijkstra(d1,1,HU(c(i),1));
    [dist2,~]=mydijkstra(d1,1,HU(c(i),2));
    if dist1<=dist2
        dd(i,1)=dist1;
    elseif dist1>dist2
        dd(i,1)=dist2;
    end
end
for i=1:k-1
    for j=1:k-1-i
        if dd(j)>dd(j+1)
            tt=c(j);
            c(j)=c(j+1);
            c(j+1)=tt;
        end
    end
end

for i=1:k
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c(i),2)];
        qi=HU(c(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c(i),1)];
        qi=HU(c(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(1,1)=dist_0(1,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c(i),1:2)];
```

```
        qi=HU(c(i),2);
    end
    if dist1>=dist2&&dist2~=0
        dist_0(1,1)=dist_0(1,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c(i),2),HU(c(i),1)];

        qi=HU(c(i),1);
    end
end

[dist1,mypath1]=mydijkstra(d1,qi,1);
a_1=size(mypath1);
dist_0(1,1)=dist_0(1,1)+dist1;
path_0=[path_0,mypath1(2:a_1(1,2))];
xx=new_position(path_0,1);yy=new_position(path_0,2);
plot(xx,yy,'-','color','b','linewidth',3);

clear path_0;
path_0=[qi];
for i=1:145
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c1(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c1(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c1(i),2)];
        qi=HU(c1(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c1(i),1)];
        qi=HU(c1(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(1,1)=dist_0(1,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c1(i),1:2)];

xx=[xx;new_position(mypath1,1);new_position(HU(i,2),1
)];

yy=[yy;new_position(mypath1,2);new_position(HU(i,2),2
)];
        qi=HU(c1(i),2);
    end
    if dist1>=dist2&&dist2~=0
```

```
        dist_0(1,1)=dist_0(1,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c1(i),2),HU(c1(i),1)];

xx=[xx;new_position(mypath2,1);new_position(HU(i,1),1
)];

yy=[yy;new_position(mypath2,2);new_position(HU(i,1),2
)];
        qi=HU(c1(i),1);
    end
end
xx=new_position(path_0,1);yy=new_position(path_0,2);
plot(xx,yy,'-','color','y');


c=randperm(95);
qi=1;
clear path_0;path_0=[qi];
for i=1:95
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c(i),2)];
        qi=HU(c(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c(i),1)];
        qi=HU(c(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(k,1)=dist_0(k,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c(i),1:2)];

        qi=HU(c(i),2);
    end
    if dist1>=dist2&&dist2~=0
        dist_0(k,1)=dist_0(k,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c(i),2),HU(c(i),1)];

        qi=HU(c(i),1);
    end
```

```
end
for k=1:w
    c=randperm(100);
    c1=[1,c+1,102];
    for t=1:102
        flag=0;
    for m=1:100
      for n=m+2:101
        if
d(c1(m),c1(n))+d(c1(m+1),c1(n+1))<d(c1(m),c1(m+1))+d(
c1(n),c1(n+1))
           c1(m+1:n)=c1(n:-1:m+1);   flag=1;
        end
      end
    end
   if flag==0
      J(k,c1)=1:102; break
   end
   end
end
J(:,1)=0; J=J/102;
for k=1:g
    A=J;
    c=randperm(w);
    for i=1:2:w
        F=2+floor(100*rand(1));
        temp=A(c(i),[F:102]);
        A(c(i),[F:102])=A(c(i+1),[F:102]);
        A(c(i+1),F:102)=temp;
    end
    by=[];
while ~length(by)
    by=find(rand(1,w)<0.1);
end
B=A(by,:);
for j=1:length(by)
   bw=sort(2+floor(100*rand(1,3)));
   B(j,:)=B(j,[1:bw(1)-
1,bw(2)+1:bw(3),bw(1):bw(2),bw(3)+1:102]);
end
   G=[J;A;B];
   [SG,ind1]=sort(G,2);
   num=size(G,1); long=zeros(1,num);
   for j=1:num
       for i=1:101
           long(j)=long(j)+d(ind1(j,i),ind1(j,i+1));
        end
    end
      [slong,ind2]=sort(long);
      J=G(ind2(1:w),:);
end
path=ind1(ind2(1),:), flong=slong(1)
xx=sj(path,1);yy=sj(path,2);
plot(xx,yy,'-o')
df=dist_0(k,1)-dist_0(k-1,1);
if df<0
    clear c_z;
    clear path_z;
    dist_z=dist_0(k,1);
    path_z=path_0;
    c_z=c;
elseif exp(-df/T_1)>=rand
    clear c_z;
    clear path_z;
    dist_z=dist_0(k,1);
    path_z=path_0;
    c_z=c;
end
T_1=T_1*at;
if T_1<e_1
    break;
end
end
xx=new_position(path_0,1);yy=new_position(path_0,2);
plot(xx,yy,'-o')
xx=new_position(path_z,1);yy=new_position(path_z,2);
plot(xx,yy,'-o')
```

Question 2

```
clc,clear
link=xlsread("links.xls");
sj0=xlsread("Attachment 1.xlsx");
loc_nodes=xlsread("Attachment 1.xlsx");
startnode=link(:,1);
endnode=link(:,2);
width=link(:,3);
HU=zeros(241,3);
HU(:,1)=startnode;
HU(:,2)=endnode;
HU(:,3)=width/2;
ZOHU=sum(HU);
ZOHU1=ZOHU(1,3);
```

```
link1=zeros(141);
for n=1:241
    link1(startnode(n),endnode(n))=1;
end
link1=link1+link1';

x=sj0(:,2); x=x(:);
y=sj0(:,3); y=y(:);
kb=5;
sj=[x y];

sj=sj*pi/180;
d=zeros(141);
d1=zeros(141);
for i=1:140
    for j=i+1:141
        d(i,j)=6370*acos(cos(sj(i,1)-
sj(j,1))*cos(sj(i,2))*cos(sj(j,2))+sin(sj(i,2))*sin(s
j(j,2)));
    end
end
d=d+d';

for i=1:140
    for j=i+1:141
        if link1(i,j)==1
            d1(i,j)=d(i,j);
        end
        if link1(i,j)==0
            d1(i,j)=inf;
        end
    end
end
d1=d1+d1';




M_PI=3.14159265358979323846;

xlab=loc_nodes(:,2);
ylab=loc_nodes(:,3);
position=[xlab ylab];
L = 6381372 * M_PI * 2;
W = L;
```

```
H = L / 2;
mill = 2.3;
n=size(position,1);
new_position=[];
figure
for i =1:n
    lon=position(i,1);
    lat=position(i,2);
    x = lon * M_PI / 180;
    y = lat * M_PI / 180;
    y1 = 1.25 * log(tan(0.25 * M_PI + 0.4 * y));

    dikaerX(i) = (W / 2) + (W / (2 * M_PI)) * x ;
    dikaerY(i) = (H / 2) - (H / (2 * mill)) * y1 ;
    new_position(i,1)=dikaerX(i);
    new_position(i,2)=dikaerY(i);

plot(dikaerX(i),dikaerY(i),'o','color','k','Markersiz
e',8)
    hold on
end

path_0=[0];
path_z=[0];
qi=1;
path_0=[qi];
dist_0=zeros(L_1,1);
dist_z=[];
dist1=[];
dist2=[];
xx=[];
yy=[];

c_z=randperm(241);
k=5;
c=c_z(1:k);
c1=c_z(k:241);
for i=1:k
    [dist1,~]=mydijkstra(d1,1,HU(c(i),1));
    [dist2,~]=mydijkstra(d1,1,HU(c(i),2));
    if dist1<=dist2
        dd(i,1)=dist1;
    elseif dist1>dist2
        dd(i,1)=dist2;
    end
end
```

```
for i=1:k-1
    for j=1:k-1-i
        if dd(j)>dd(j+1)
            tt=c(j);
            c(j)=c(j+1);
            c(j+1)=tt;
        end
    end
end

for i=1:k
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c(i),2)];
        qi=HU(c(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c(i),1)];
        qi=HU(c(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(1,1)=dist_0(1,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c(i),1:2)];

        qi=HU(c(i),2);
    end
    if dist1>=dist2&&dist2~=0
        dist_0(1,1)=dist_0(1,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c(i),2),HU(c(i),1)];

        qi=HU(c(i),1);
    end
end

[dist1,mypath1]=mydijkstra(d1,qi,1);
a_1=size(mypath1);
dist_0(1,1)=dist_0(1,1)+dist1;
path_0=[path_0,mypath1(2:a_1(1,2))];
xx=new_position(path_0,1);yy=new_position(path_0,2);
plot(xx,yy,'-','color','b','linewidth',3);
```

```
clear path_0;
path_0=[qi];
for i=1:145
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c1(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c1(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c1(i),2)];
        qi=HU(c1(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c1(i),1)];
        qi=HU(c1(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(1,1)=dist_0(1,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c1(i),1:2)];

xx=[xx;new_position(mypath1,1);new_position(HU(i,2),1
)];

yy=[yy;new_position(mypath1,2);new_position(HU(i,2),2
)];
        qi=HU(c1(i),2);
    end
    if dist1>=dist2&&dist2~=0
        dist_0(1,1)=dist_0(1,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c1(i),2),HU(c1(i),1)];

xx=[xx;new_position(mypath2,1);new_position(HU(i,1),1
)];

yy=[yy;new_position(mypath2,2);new_position(HU(i,1),2
)];
        qi=HU(c1(i),1);
    end
end
xx=new_position(path_0,1);yy=new_position(path_0,2);
plot(xx,yy,'-','color','y');


c=randperm(95);
qi=1;
```

```
clear path_0;path_0=[qi];
for i=1:95
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c(i),2)];
        qi=HU(c(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c(i),1)];
        qi=HU(c(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(k,1)=dist_0(k,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c(i),1:2)];

        qi=HU(c(i),2);
    end
    if dist1>=dist2&&dist2~=0
        dist_0(k,1)=dist_0(k,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c(i),2),HU(c(i),1)];

        qi=HU(c(i),1);
    end
end
for k=1:w
    c=randperm(100);
    c1=[1,c+1,102];
    for t=1:102
        flag=0;
    for m=1:100
      for n=m+2:101
        if
d(c1(m),c1(n))+d(c1(m+1),c1(n+1))<d(c1(m),c1(m+1))+d(
c1(n),c1(n+1))
            c1(m+1:n)=c1(n:-1:m+1);   flag=1;
        end
      end
    end
   if flag==0
      J(k,c1)=1:102; break
   end
   end
end
J(:,1)=0; J=J/102;
for k=1:g
    A=J;
    c=randperm(w);
    for i=1:2:w
        F=2+floor(100*rand(1));
        temp=A(c(i),[F:102]);
        A(c(i),[F:102])=A(c(i+1),[F:102]);
        A(c(i+1),F:102)=temp;
    end
    by=[];
while ~length(by)
    by=find(rand(1,w)<0.1);
end
B=A(by,:);
for j=1:length(by)
   bw=sort(2+floor(100*rand(1,3)));
   B(j,:)=B(j,[1:bw(1)-
1,bw(2)+1:bw(3),bw(1):bw(2),bw(3)+1:102]);
end
   G=[J;A;B];
   [SG,ind1]=sort(G,2);
   num=size(G,1); long=zeros(1,num);
   for j=1:num
       for i=1:101
           long(j)=long(j)+d(ind1(j,i),ind1(j,i+1));
       end
   end
     [slong,ind2]=sort(long);
     J=G(ind2(1:w),:);
end
path=ind1(ind2(1),:), flong=slong(1)
xx=sj(path,1);yy=sj(path,2);
plot(xx,yy,'-o')
df=dist_0(k,1)-dist_0(k-1,1);
if df<0
    clear c_z;
    clear path_z;
    dist_z=dist_0(k,1);
    path_z=path_0;
    c_z=c;
elseif exp(-df/T_1)>=rand
    clear c_z;
    clear path_z;
    dist_z=dist_0(k,1);
    path_z=path_0;
    c_z=c;
end
T_1=T_1*at;
if T_1<e_1
    break;
end
end
xx=new_position(path_0,1);yy=new_position(path_0,2);
plot(xx,yy,'-o')
xx=new_position(path_z,1);yy=new_position(path_z,2);
plot(xx,yy,'-o')
clear
clc

link=xlsread("links.xls");
startnode=link(:,1);
endnode=link(:,2);
width=link(:,3);
HU=zeros(241,3);
sj0=xlsread("Attachment 1.xlsx");

link1=zeros(141)
for n=1:241
    link1(startnode(n),endnode(n))=1;
end
link1=link1+link1';
x_w=sj0(:,2); x_w=x_w(:);
y_w=sj0(:,3); y_w=y_w(:);
kb=5;
sj=[x_w y_w];
sj=sj*pi/180;
d=zeros(141);
d1=zeros(141);
for i=1:140
  for j=i+1:141
      d(i,j)=6370*acos(cos(sj(i,1)-
sj(j,1))*cos(sj(i,2))*cos(sj(j,2))+sin(sj(i,2))*sin(s
j(j,2)));
  end
end
d=d+d';
for i=1:140
  for j=i+1:141
      if link1(i,j)==1
```

```
          d1(i,j)=d(i,j);
      end
      if link1(i,j)==0
          d1(i,j)=inf;
      end
  end
end
d1=d1+d1';
M_PI=3.14159265358979323846;
xlab1=loc_nodes(:,2);
ylab1=loc_nodes(:,3);
position=[xlab1 ylab1];
L = 6381372 * M_PI * 2;
W = L;
H = L / 2;
mill = 2.3;
n=size(position,1);
new_position=[];
figure
for i =1:n
    lon=position(i,1);
    lat=position(i,2);
    x = lon * M_PI / 180;
    y_j = lat * M_PI / 180;
    y1 = 1.25 * log(tan(0.25 * M_PI + 0.4 * y_j));

    dikaerX(i) = (W / 2) + (W / (2 * M_PI)) * x ;
    dikaerY(i) = (H / 2) - (H / (2 * mill)) * y1 ;
    new_position(i,1)=dikaerX(i);
    new_position(i,2)=dikaerY(i);

    hold on
end
loc_nodes=xlsread("snow.xls");
SNSIZE=loc_nodes(:,3)/100;
xlab1=loc_nodes(:,1);
ylab1=loc_nodes(:,2);
position=[xlab1 ylab1];
L = 6381372 * M_PI * 2;
W = L;
H = L / 2;
mill = 2.3;
n=size(position,1);
new_position1=[];

for i =1:n
```

```
    lon=position(i,1);
    lat=position(i,2);
    x = lon * M_PI / 180;
    y_j = lat * M_PI / 180;
    y1 = 1.25 * log(tan(0.25 * M_PI + 0.4 * y_j));

    dikaerX1(i) = (W / 2) + (W / (2 * M_PI)) * x ;
    dikaerY1(i) = (H / 2) - (H / (2 * mill)) * y1 ;
    new_position1(i,1)=dikaerX1(i);
    new_position1(i,2)=dikaerY1(i);

    hold on
end
x=[dikaerX,dikaerX1];
y_j=[dikaerY,dikaerY1];
for i=1:171
    if i<=141
    plot(x(i),y_j(i),'.','color','k','Markersize',9)
    elseif i<=171

plot(x(i),y_j(i),'o','color','b','Markersize',SNSIZE(
i-141))
    end
end
C=zeros(30,1);
for m=1:30
    for n=1:141
    if x_w(n)==xlab1(m)&&y_w(n)==ylab1(m)
        C(m,1)=n;
    end
    end
end
disttt=zeros(30,1);
for i=1:30
[dist1,mypath1]=mydijkstra(d1,1,C(25));
xx=new_position(mypath1,1);yy=new_position(mypath1,2)
;
plot(xx,yy,'-','color','r');
disttt(i,1)=dist1;
end
```

Question 3、4

```
clc,clear
```

```
link=xlsread("links.xls");
sj0=xlsread("Attachment 1.xlsx");
loc_nodes=xlsread("Attachment 1.xlsx");
startnode=link(:,1);
endnode=link(:,2);
width=link(:,3);
HU=zeros(241,3);
HU(:,1)=startnode;
HU(:,2)=endnode;
HU(:,3)=width/2;
ZOHU=sum(HU);
ZOHU1=ZOHU(1,3);
link1=zeros(141);
for n=1:241
    link1(startnode(n),endnode(n))=1;
end
link1=link1+link1';

x=sj0(:,2); x=x(:);
y=sj0(:,3); y=y(:);
kb=5;
sj=[x y];

sj=sj*pi/180;
d=zeros(141);
d1=zeros(141);
for i=1:140
    for j=i+1:141
        d(i,j)=6370*acos(cos(sj(i,1)-
sj(j,1))*cos(sj(i,2))*cos(sj(j,2))+sin(sj(i,2))*sin(s
j(j,2)));
    end
end
d=d+d';

for i=1:140
    for j=i+1:141
        if link1(i,j)==1
            d1(i,j)=d(i,j);
        end
        if link1(i,j)==0
            d1(i,j)=inf;
        end
    end
end
d1=d1+d1';
```

```
a=xlsread("Attachment 2.xlsx");
startnode=a(:,1);
endnode=a(:,2);
greed=a(:,3);
for i=1:length(startnode)
    sx(i)=dikaerX(startnode(i));
    sy(i)=dikaerY(startnode(i));
    ex(i)=dikaerX(endnode(i));
    ey(i)=dikaerY(endnode(i));
    hold on
    if greed(i)==6

plot([sx(i),ex(i)],[sy(i),ey(i)],'color','g','linewid
th',3)
    end
    if greed(i)==5

plot([sx(i),ex(i)],[sy(i),ey(i)],'color','m','linewid
th',3)
    end
    if greed(i)==4

plot([sx(i),ex(i)],[sy(i),ey(i)],'color','b','linewid
th',3)
    end
    if greed(i)==3

plot([sx(i),ex(i)],[sy(i),ey(i)],'color','r','linewid
th',3)
    end
    if greed(i)==2

plot([sx(i),ex(i)],[sy(i),ey(i)],'color','c','linewid
th',3)
    end
    if greed(i)==1

plot([sx(i),ex(i)],[sy(i),ey(i)],'color','k','linewid
th',3)
    end
end
M_PI=3.14159265358979323846;

xlab=loc_nodes(:,2);
ylab=loc_nodes(:,3);
position=[xlab ylab];
```

```
L = 6381372 * M_PI * 2;
W = L;
H = L / 2;
mill = 2.3;
n=size(position,1);
new_position=[];
figure
for i =1:n
    lon=position(i,1);
    lat=position(i,2);
    x = lon * M_PI / 180;
    y = lat * M_PI / 180;
    y1 = 1.25 * log(tan(0.25 * M_PI + 0.4 * y));

    dikaerX(i) = (W / 2) + (W / (2 * M_PI)) * x ;
    dikaerY(i) = (H / 2) - (H / (2 * mill)) * y1 ;
    new_position(i,1)=dikaerX(i);
    new_position(i,2)=dikaerY(i);

plot(dikaerX(i),dikaerY(i),'o','color','k','Markersiz
e',8)
    hold on
end

path_0=[0];
path_z=[0];
qi=1;
path_0=[qi];
dist_0=zeros(L_1,1);
dist_z=[];
dist1=[];
dist2=[];
xx=[];
yy=[];

c_z=randperm(241);
k=5;
c=c_z(1:k);
c1=c_z(k:241);
for i=1:k
    [dist1,~]=mydijkstra(d1,1,HU(c(i),1));
    [dist2,~]=mydijkstra(d1,1,HU(c(i),2));
    if dist1<=dist2
        dd(i,1)=dist1;
    elseif dist1>dist2
        dd(i,1)=dist2;
    end
end
for i=1:k-1
    for j=1:k-1-i
        if dd(j)>dd(j+1)
            tt=c(j);
            c(j)=c(j+1);
            c(j+1)=tt;
        end
    end
end

for i=1:k
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c(i),2)];
        qi=HU(c(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c(i),1)];
        qi=HU(c(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(1,1)=dist_0(1,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c(i),1:2)];

        qi=HU(c(i),2);
    end
    if dist1>=dist2&&dist2~=0
        dist_0(1,1)=dist_0(1,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c(i),2),HU(c(i),1)];

        qi=HU(c(i),1);
    end
end

[dist1,mypath1]=mydijkstra(d1,qi,1);
a_1=size(mypath1);
dist_0(1,1)=dist_0(1,1)+dist1;
path_0=[path_0,mypath1(2:a_1(1,2))];
xx=new_position(path_0,1);yy=new_position(path_0,2);
```

```
plot(xx,yy,'-','color','b','linewidth',3);

clear path_0;
path_0=[qi];
for i=1:145
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c1(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c1(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c1(i),2)];
        qi=HU(c1(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c1(i),1)];
        qi=HU(c1(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(1,1)=dist_0(1,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c1(i),1:2)];

xx=[xx;new_position(mypath1,1);new_position(HU(i,2),1
)];

yy=[yy;new_position(mypath1,2);new_position(HU(i,2),2
)];
        qi=HU(c1(i),2);
    end
    if dist1>=dist2&&dist2~=0
        dist_0(1,1)=dist_0(1,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c1(i),2),HU(c1(i),1)];

xx=[xx;new_position(mypath2,1);new_position(HU(i,1),1
)];

yy=[yy;new_position(mypath2,2);new_position(HU(i,1),2
)];
        qi=HU(c1(i),1);
    end
end
xx=new_position(path_0,1);yy=new_position(path_0,2);
plot(xx,yy,'-','color','y');
```

```
c=randperm(95);
qi=1;
clear path_0;path_0=[qi];
for i=1:95
    [dist1,mypath1]=mydijkstra(d1,qi,HU(c(i),1));
    [dist2,mypath2]=mydijkstra(d1,qi,HU(c(i),2));
    a_1=size(mypath1);
    a_2=size(mypath2);
    if dist1==0
        path_0=[path_0,HU(c(i),2)];
        qi=HU(c(i),2);
    end
    if dist2==0
        path_0=[path_0,HU(c(i),1)];
        qi=HU(c(i),1);
    end
    if dist1<dist2&&dist1~=0
        dist_0(k,1)=dist_0(k,1)+dist1;
        path_0=[path_0,mypath1(2:a_1(1,2)-
1),HU(c(i),1:2)];

        qi=HU(c(i),2);
    end
    if dist1>=dist2&&dist2~=0
        dist_0(k,1)=dist_0(k,1)+dist2;
        path_0=[path_0,mypath2(2:a_2(1,2)-
1),HU(c(i),2),HU(c(i),1)];

        qi=HU(c(i),1);
    end
end
for k=1:w
    c=randperm(100);
    c1=[1,c+1,102];
    for t=1:102
        flag=0;
    for m=1:100
      for n=m+2:101
        if
d(c1(m),c1(n))+d(c1(m+1),c1(n+1))<d(c1(m),c1(m+1))+d(
c1(n),c1(n+1))
           c1(m+1:n)=c1(n:-1:m+1);  flag=1;
        end
      end
    end
   if flag==0
```

```
      J(k,c1)=1:102; break
   end
   end
end
J(:,1)=0; J=J/102;
for k=1:g
    A=J;
    c=randperm(w);
    for i=1:2:w
        F=2+floor(100*rand(1));
        temp=A(c(i),[F:102]);
        A(c(i),[F:102])=A(c(i+1),[F:102]);
        A(c(i+1),F:102)=temp;
    end
    by=[];
while ~length(by)
    by=find(rand(1,w)<0.1);
end
B=A(by,:);
for j=1:length(by)
   bw=sort(2+floor(100*rand(1,3)));
   B(j,:)=B(j,[1:bw(1)-
1,bw(2)+1:bw(3),bw(1):bw(2),bw(3)+1:102]);
end
   G=[J;A;B];
   [SG,ind1]=sort(G,2);
   num=size(G,1); long=zeros(1,num);
   for j=1:num
       for i=1:101
           long(j)=long(j)+d(ind1(j,i),ind1(j,i+1));
       end
   end
     [slong,ind2]=sort(long);
     J=G(ind2(1:w),:);
end
path=ind1(ind2(1),:), flong=slong(1)
xx=sj(path,1);yy=sj(path,2);
plot(xx,yy,'-o')
df=dist_0(k,1)-dist_0(k-1,1);
if df<0
    clear c_z;
    clear path_z;
    dist_z=dist_0(k,1);
    path_z=path_0;
    c_z=c;
elseif exp(-df/T_1)>=rand
```

```
    clear c_z;
    clear path_z;
    dist_z=dist_0(k,1);
    path_z=path_0;
    c_z=c;
end
T_1=T_1*at;
if T_1<e_1
    break;
end
end
xx=new_position(path_0,1);yy=new_position(path_0,2);
plot(xx,yy,'-o')
xx=new_position(path_z,1);yy=new_position(path_z,2);
plot(xx,yy,'-o')
clear
clc

link=xlsread("links.xls");
startnode=link(:,1);
endnode=link(:,2);
width=link(:,3);
HU=zeros(241,3);
sj0=xlsread("Attachment 1.xlsx");

link1=zeros(141)
for n=1:241
    link1(startnode(n),endnode(n))=1;
end
link1=link1+link1';
x_w=sj0(:,2); x_w=x_w(:);
y_w=sj0(:,3); y_w=y_w(:);
kb=5;
sj=[x_w y_w];
sj=sj*pi/180;
d=zeros(141);
d1=zeros(141);
for i=1:140
  for j=i+1:141
      d(i,j)=6370*acos(cos(sj(i,1)-
sj(j,1))*cos(sj(i,2))*cos(sj(j,2))+sin(sj(i,2))*sin(s
j(j,2)));
  end
end
d=d+d';
for i=1:140
```

```
  for j=i+1:141
      if link1(i,j)==1
          d1(i,j)=d(i,j);
      end
      if link1(i,j)==0
          d1(i,j)=inf;
      end
  end
end
d1=d1+d1';
M_PI=3.14159265358979323846;
xlab1=loc_nodes(:,2);
ylab1=loc_nodes(:,3);
position=[xlab1 ylab1];
L = 6381372 * M_PI * 2;
W = L;
H = L / 2;
mill = 2.3;
n=size(position,1);
new_position=[];
figure
for i =1:n
    lon=position(i,1);
    lat=position(i,2);
    x = lon * M_PI / 180;
    y_j = lat * M_PI / 180;
    y1 = 1.25 * log(tan(0.25 * M_PI + 0.4 * y_j));

    dikaerX(i) = (W / 2) + (W / (2 * M_PI)) * x ;
    dikaerY(i) = (H / 2) - (H / (2 * mill)) * y1 ;
    new_position(i,1)=dikaerX(i);
    new_position(i,2)=dikaerY(i);

    hold on
end
loc_nodes=xlsread("snow.xls");
SNSIZE=loc_nodes(:,3)/100;
xlab1=loc_nodes(:,1);
ylab1=loc_nodes(:,2);
position=[xlab1 ylab1];
L = 6381372 * M_PI * 2;
W = L;
H = L / 2;
mill = 2.3;
n=size(position,1);
new_position1=[];
```

```
for i =1:n
    lon=position(i,1);
    lat=position(i,2);
    x = lon * M_PI / 180;
    y_j = lat * M_PI / 180;
    y1 = 1.25 * log(tan(0.25 * M_PI + 0.4 * y_j));

    dikaerX1(i) = (W / 2) + (W / (2 * M_PI)) * x ;
    dikaerY1(i) = (H / 2) - (H / (2 * mill)) * y1 ;
    new_position1(i,1)=dikaerX1(i);
    new_position1(i,2)=dikaerY1(i);

    hold on
end
x=[dikaerX,dikaerX1];
y_j=[dikaerY,dikaerY1];
for i=1:171
    if i<=141
    plot(x(i),y_j(i),'.','color','k','Markersize',9)
    elseif i<=171

plot(x(i),y_j(i),'o','color','b','Markersize',SNSIZE(
i-141))
    end
end
C=zeros(30,1);
for m=1:30
    for n=1:141
    if x_w(n)==xlab1(m)&&y_w(n)==ylab1(m)
        C(m,1)=n;
    end
    end
end
disttt=zeros(30,1);
for i=1:30
[dist1,mypath1]=mydijkstra(d1,1,C(25));
xx=new_position(mypath1,1);yy=new_position(mypath1,2)
;
plot(xx,yy,'-','color','r');
disttt(i,1)=dist1;
end
```