Problem Chosen	2024 MCM/ICM Summary Sheet	Team Control Number
B		223

Enhancing Taxi Route Planning: A Comprehensive Analysis of ISPL

Summary

Taxi companies need to consider user preferences reflected in historical path data because the best route during a taxi ride may not necessarily be the shortest route. It is important to take into account both the distance and user preferences in order to determine the optimal route. Assigning weights to paths based on ambiguous user preferences and planning the shortest path under corresponding conditions belongs to a type of an inverse shortest path length problem(ISPL). To address this problem, we introduce edge PageRank to measure road weights and utilize simulated annealing algorithm to find the maximum SIM value.

For the first task, we prove by contradiction that the SIM value in example 1 cannot be equal to 1. Additionally, we attempted to assign weights to each edge, **resulting in a SIM value of 0.75**.

For the second task, we first preprocessed the data in Case 1. We placed the data points on a map of Beijing to observe their distribution characteristics and removed duplicate edges and identified edges that did not appear in any of the paths. We then investigated the number of times each edge was traversed in the paths and analyzed the distribution characteristics of the data.Then, we used **Dijkstra's algorithm** to find the shortest path length for each path. We sorted the paths in ascending order based on their starting and ending points. By processing all paths with the same starting point at once using an array, we significantly reduced the time complexity. Finally, we calculated that **the SIM value for Case 1 is 0.248971**.

For the third task, in our initial model, we used **road length, road importance (measured by edge PageRank), and straight-line distance** to measure road weights. We calculated the straight-line distance based on the latitude and longitude of the starting and ending points. We calculated **the maximum SIM value to be 0.273278**. We then conducted further analysis and found that the straight-line distance had little impact on the SIM value. However, we discovered a significant relationship **between the data point's index and its location**. Therefore, we decided to optimize our model by excluding the straight-line distance, introduceing the edge index into the calculation formula for edge PageRank and adding a constant. Ultimately, we determined **several parameters, denoted as 26.27088, 0.01172792, and 0.256358**, to calculate the edge weights. It results in **a maximum SIM value of 0.298223**.

For the fourth task,we first compared the optimization rates of the two models in terms of SIM values obtained compared to using length as edge weights. Next, we plotted scatter plots **comparing the distribution differences between edge weights and edge lengths**, and analyzed the impact of introducing PageRank into the weight calculation.

Finally, we conducted **sensitivity tests** on the resulting model. We selected the parameters **strides and initial and final temperatures** for testing and found that the model exhibited good stability for different values of these parameters.

Keywords: Route Planning Inverse Shortest Path Length Simulated Annealing Algorithm

Contents

1 Introduction

1.1 Problem Background

With the emergence of ride-hailing companies like Didi Chuxing and Uber, people are increasingly inclined to travel by taxi. Providing users with suitable driving routes based on their preferences is an important aspect of enhancing the user experience. Historical data indicates that the shortest distance or travel time may not necessarily be the most appropriate route. The user's preferences seem to be influenced by various other factors.

Due to the ambiguity of evaluation metrics, it is challenging to find a universally applicable evaluation method for automatically selecting the optimal path. Therefore, we need to extract user preference information from historical data and give the selection weights for different routes. It will be used for future route selection, which is of importance for the development of taxi companies.

1.2 Restatement of the Problem

Having understood the problem, we figure out the following tasks:

- *Task 1*: According to the computational method of SIM, we aim to determine **the maximum value of SIM in Example 1** and **find a set of weights for each edge in the directed weighted graph**.
- *Task 2*: Based on the data file given in the question,we **treat the length of the edges as the weights of the roads** without considering historical data and user preferences. And we need to **calculate the SIM value in Case 1**.
- *Task 3*: We need to design a model that can **determine the weights of each edge in any given graph** in order to maximize the SIM value. Once the model is designed, we will apply it to Case 1 mentioned and showcase the maximum SIM value.
- *Task 4*: The weight values provided by our designed model should differ from the edge lengths. We need to explain **the difference between the two** and demonstrate through how the weight values we provide, compared to the edge lengths, offer advantages in terms of path selection.

1.3 Previous Reserach

This problem essentially belongs to ISPL problem in inverse combinatorial optimization. It requires us to optimize and determine accurate edge weights from partially defined information such as historical data. The objective is to ensure that the maximum number of paths coincide with the shortest path, resulting in the maximum SIM value. Prior to this, a considerable number of scholars have conducted research in this area.

Burton and Toint were among the pioneering researchers in the study of the inverse shortest path problem. They specialized the binary quadratic programming method proposed by Goldfarb and Idnani in 1983. And subsequently, the problem was regarded as a linear programming model. Research revealed that the problem could be transformed into a minimum-cost cycle problem and could not be solved using polynomial algorithms. Cai and Yang addressed the ISPP problem in 1994, and since

then, several algorithms have been proposed, including those based on linear and bilinear programming models (Ahmed and Guan, 2004) and genetic algorithms (António Leitão, Adriano Vinhas, Penousal Machado, Francisco Câmara Pereira, 2017) to solve the ISPL problem. Therefore, based on the reference to previous methods, we ultimately chose to use the simulated annealing algorithm to solve this problem.

1.4 Our Work

For convenience, we draw a flow chart to better represent our work:

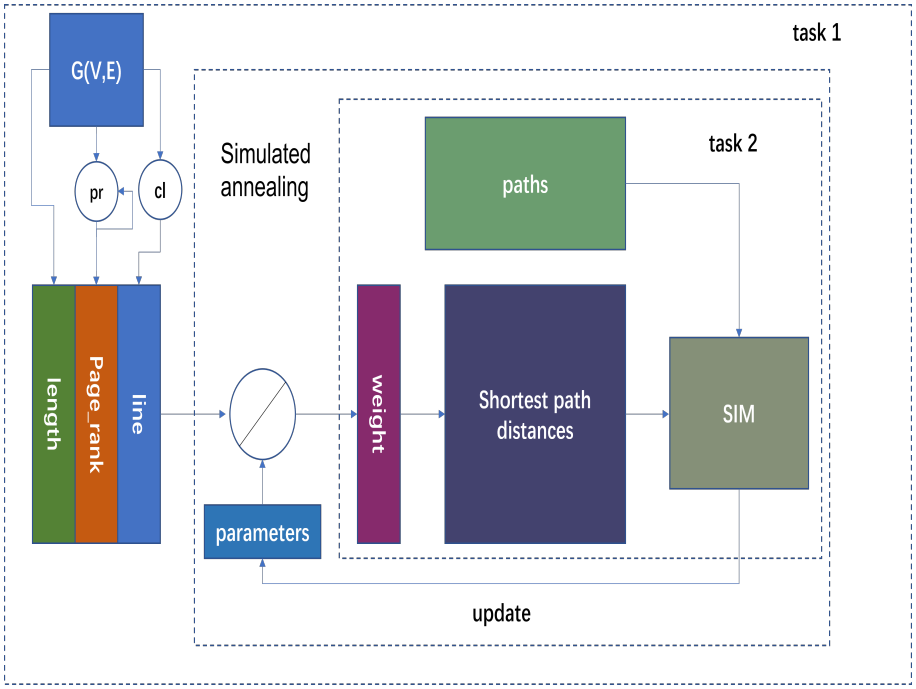


Figure 1: Our Mind Map for tasks

- Based on the given information that the SIM value in Example 1 is used to determine whether four paths belong to the shortest paths, it can only take five values: 0, 0.25, 0.5, 0.75, and 1. We begin by proving that in Example 1, the SIM value cannot be equal to 1. Secondly, we find a set of edge weights that result in a SIM value of 0.75 and verify it using a computer program. Therefore, we conclude that **the maximum value for the SIM value is 0.75**.

- We utilized the Dijkstra's algorithm to establish a model for calculating the SIM value of a known weighted graph. We constructed a weighted directed graph using the provided data of points and edges in Case 1. Then, for the given choices of the starting and ending points for path selection, we computed the shortest path and analyzed whether the length of each given path was equal to the minimum length. We also calculated the percentage of paths that satisfied this condition.Based on the analysis we conducted for Case 1, we obtained a **SIM value of 0.248971**.

- We first proposed factors to measure the road weight: **road length, road importance, and the straight-line distance**. Drawing inspiration from PageRank, we introduced **the edge PageRank**

value to determine the importance of roads. We calculated the straight-line distance of roads based on the weighted sum of the squared latitude and longitude differences between the road's starting and ending points, considering the impact of taking detours on customer preferences. Based on this, we designed a model with **simulated annealing algorithm** and substituted the data from Case 1 into the models to determine the parameters. In the model, we explicitly considered the straight-line distance between the starting and ending points, **with the maximum SIM value of 0.273278**. Then, we optimized the calculation of edge PageRank by incorporating the straight-line distance and the objective positions of the starting and ending points. We then obtained **the maximum SIM values of 0.298223 for Case 1**.

- We calculated the weights of each edge and used two different models to **increase the original SIM value by 9.76298% and 19.7822% respectively**.We plotted the edge weights against the edge lengths in a scatter plot and analyzed the corresponding relationship between them, proposing possible reasons for it. We also analyzed the relationship between **the indicator of road importance (PageRank) and edge length**, and examined how it influences the final edge weights.

2 Assumptions and Justifications

- ***Assumption 1*:The lengths of the edges provided in the data file are based on actual measurements or reliable data, rather than estimated values or simulated results.**
  *Explanation:*The data in the file consists of the latitude and longitude coordinates of different locations in Beijing, along with the directed distances between two points. The data has a high level of accuracy and provides a realistic simulation of real-world scenarios.

- ***Assumption 2*:Historical paths can represent the path selection preferences of users over a relatively long period of time, and their trends are representative.**
  *Explanation:*Historical data is meaningful only when it reflects long-term and stable user preference.Representative selection tendencies can assist the model in better uncovering user preference information.

- ***Assumption 3*:The region under consideration is a plane rather than a curved surface, and the straight-line distance between two locations is solely determined by their longitude and latitude.**
  *Explanation:*Because the straight-line distance between two points on a sphere cannot be directly calculated based on their longitude and latitude,We need to find a simplified way of calculation.However, in the given scenario where the area is relatively small, it is possible to approximate it as a plane and handle it accordingly.

3 Notations

Symbol	Definition
SIM	Value of the SIM
a_i	Evaluation indicator
α	Sum of the shortest paths
A	Sum of all paths
u	The starting point of the edge
v	The endpoint of the edge
$PR_e^{\gamma}(G)$	The PageRank of the edge e in graph G
$dist_e$	The straight-line distance of the edge e
$length_e$	The length of the edge e
ω_e	The weight of the edge e

4 Model

4.1 Designing Weights to Maxmize the SIM in Example 1

In the first task, we need to address two issues: what is the maximum SIM in Example 1, and what are the edge weights that result in the maximum SIM. Taking into account that Example 1 only considers four paths with two sets of starting and ending points, namely $(1, 7)$ and $(1, 9)$, **SIM can only be one of the values 0, 0.25, 0.5, 0.75, or 1**. In the given example, there are already cases where SIM is 0.25 and 0.5. Therefore, we first aim to determine whether SIM can be 1. If it is possible, then we will proceed to explore the edge weights that meet this condition. Otherwise, we consider **the case where SIM is 0.75**.

We use a **proof by contradiction** to demonstrate that SIM cannot be equal to 1. As shown in Figure 1, it illustrates the various points and partial edges in Example 1. The letters labeled beside some of the edges represent the lengths of those directed edges.

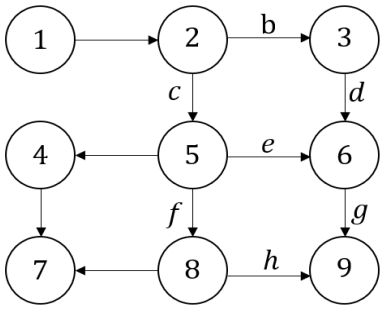


Figure 2: Some Edge Weights in Example 1

According to the conditions, if the SIM equals to 1, then for paths $p3$ and $p4$, both paths are the shortest path of (1, 9). The relation between the lengths of the two paths is as follows:

$$\begin{cases} c+e+g=b+d+e+f+h \\ c+f+h \geq b+d+e+f+h \\ b+d+g \geq c+e+g \end{cases}$$

We simplify the above equations, and the range of possible values for e can be determined as:

$$b+d-e \geq c \geq b+d+e$$

To ensure the existence of positive integer solutions for c in the given inequality, we have the following condition for e:

$$e \leq 0$$

The inequality above **contradicts the condition stated in the problem that "all path weights are positive integers."** Therefore, the value of the SIM cannot equal to 1.

Since the SIM cannot equal to 1, we begin to seek for cases where the SIM equals to 0.75. We set the weights of the directed edges along paths $p1$, $p2$, and $p3$ to 1, and adjust the weights of the edges $1 \to 4$, $2 \to 3$, and $8 \to 9$ in order to achieve $D(1,7) = D(1,9) = 4$. Thus, we ensure that $p1$, $p2$, and $p3$ are the shortest paths corresponding to their respective starting and ending points, while $p4$ is not the shortest path. These adjustments result in a SIM value of 0.75. The results are as follows:

Table 1: The Results of the Problem 1

v_i	1	1	2	2	3	4	4	5	5	5	6	6	7	8	8	9
v_j	2	4	3	5	6	5	7	4	6	8	5	9	8	7	9	8
$w(v_i,v_j)$	1	4	4	1	1	1	1	1	1	1	1	1	1	1	4	1

4.2 Data Preprocessing for Case 1

Due to the large number of data points and edges in Case 1, it is not feasible to obtain the optimal solution through trial and error. Therefore, we need to use a computer program for calculation. Before that, we need to preprocess the data and visualize it in order to better analyze the data based on real-world circumstances.

We show all the data points provided in the question on the map of Beijing. The results are as follows:

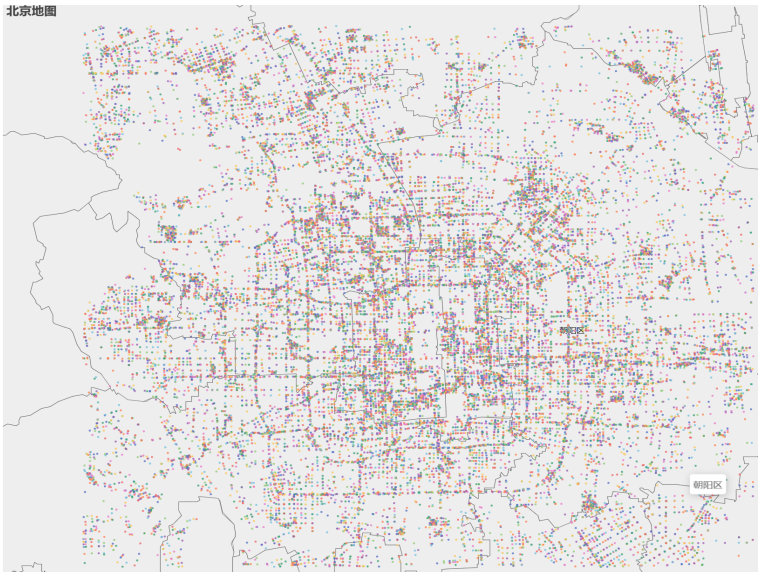
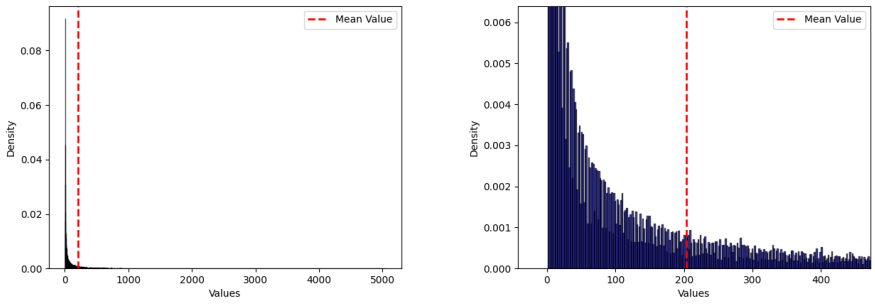


Figure 3: All Vertices Shown on the Map of Beijing

We discovered that many points are located along the tracks of public transportation lines and residential areas, which aligns with our general knowledge.

We then proceed to process the given edge values in the question. We noticed that among the provided edges, there are some instances where the length of certain edges **appears twice and is inconsistent**. We believe that this portion of the data is erroneous. Therefore, when using the data in practice, we only consider the length of the edge as it appears for the first time and remove any other erroneous data.There were **a total of 72156 edge length data** in the question, and after processing, we are **left with 71461 valid data**.

We also analyzed the given path data in the question. For each path, we incremented the usage count of each edge by 1. We traversed all the paths, summed up the usage counts of all edges, and plotted a graph showing the proportion of edge usage counts. The figure is shown below, where the vertical axis represents the proportion and the horizontal axis represents the usage count of the edges:



(a) Probability Distribution Graph of All Usage Counts

(b) Probability Distribution Graph of Partial Usage Counts

Figure 4: Histogram of Usage Counts Frequency Distribution

We found that the majority of edges have usage counts concentrated within 50 times or less. However, there is still a significant portion of edges that have usage counts exceeding 200. We believe that the more an edge is used, the more preferred it is by customers. Consequently, the importance of such an edge also increases. This will be taken into account because it influences our calculation of the road importance indicator.

Finally,we use the valid vertices and edges to make an gragh G for case 1. Now we begin to use the graph G to construct models.

4.3 Calculating the SIM Value of Case 1

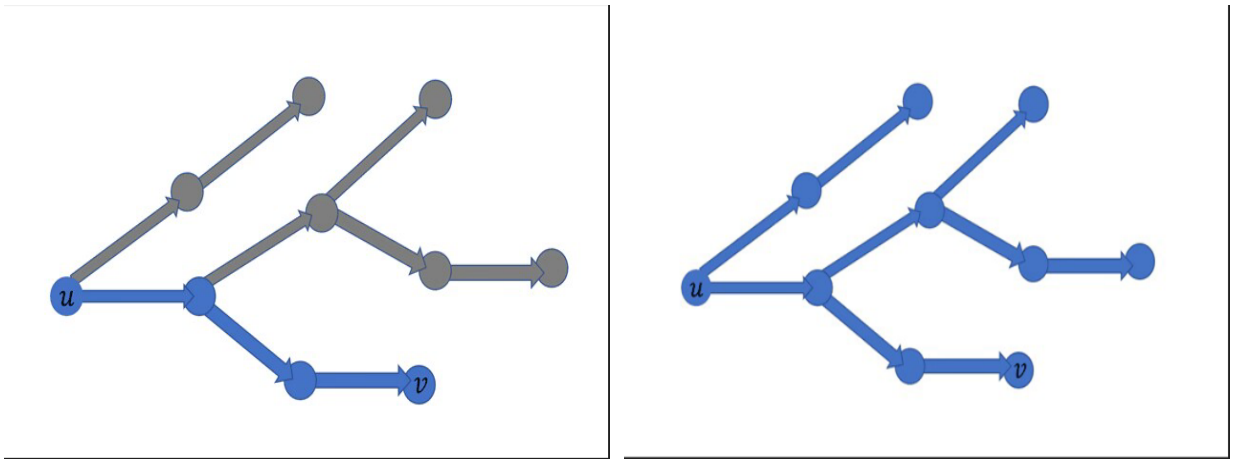
To calculate the value of SIM in Case 1, we utilize **the Dijkstra's algorithm** to compute the shortest path for each pair of starting and ending points. Since the SIM value requires counting the number of shortest paths and the total number of paths, we use a program to calculate the total number of paths visited. While traversing each path, we also record its length. Then, in combination with the Dijkstra's algorithm, we verify whether the path is indeed the shortest path for the given pair of starting and ending points.

Dijkstra's algorithm is an algorithm used to find the shortest path from a single source node in a weighted directed or undirected graph. The goal of the algorithm is to find the shortest paths from the source node to all other nodes. It starts from the source node and gradually expands to other nodes by continuously selecting the node with the current shortest path to update the shortest paths and distances.

The time complexity of the algorithm depends on the size of the graph and is typically $O(V^2)$, where V is the number of nodes in the graph. When using a priority queue data structure to optimize the algorithm implementation, the time complexity can be reduced to $O((V + E)logV)$, where E is the number of edges in the graph. In our actual computation process, we found that the former time complexity is too high, so we adopted **the priority queue optimization algorithm** to reduce the time complexity.

We know that using Dijkstra's algorithm, we can calculate the shortest paths from a single starting point to all other points in one go. However, the paths provided in the question are in a disorderly sequence. In order to further reduce the time complexity, we **sorted the paths in the file in ascending order based on the numbers of the starting and ending points**. Moreover, for all paths with the same starting point, we only called the algorithm once and used an array to store the shortest path lengths from that point to all other points. This significantly reduced the time complexity.

For example, after sorting the starting and ending point numbers in ascending order, we found a total of 64 paths with a starting point of 2. If we use the optimized algorithm, we only need to call the algorithm once and then access the generated array 64 times to calculate the shortest paths for these 64 sets of edges. Otherwise, if we had to re-call the algorithm for each path separately, the time consumption would be significant. Our basic approch is as follows:



(a) Before Optimization (b) After Optimization

Figure 5: Basic approach of Our Optimization to the Dijkstra Algorithm

When we iterate over each i-th path, we denote whether this path is the shortest path between the starting point and ending point of the group as a_i. If it is, the value is 1; otherwise, it is 0. This is represented by the following equation:

$$a_i = \begin{cases} 1, \text{it is the shortest path} \\ 0, \text{otherwise} \end{cases}$$

We define α **as the number of paths in the provided file's paths that satisfy the shortest path condition**. The calculation formula for α is as follows:

$$\alpha = \sum_{i=1}^{n} a_i \tag{1}$$

So, the value of SIM can be obtained from the following equation:

$$SIM = \frac{\alpha}{A} \tag{2}$$

We define A **as the sum of all the numbers of paths**. And we import the data that has been processed into the program and obtain the changes in SIM as the program sequentially imports paths according to the order of the provided files in the question. The change curve is as follows:

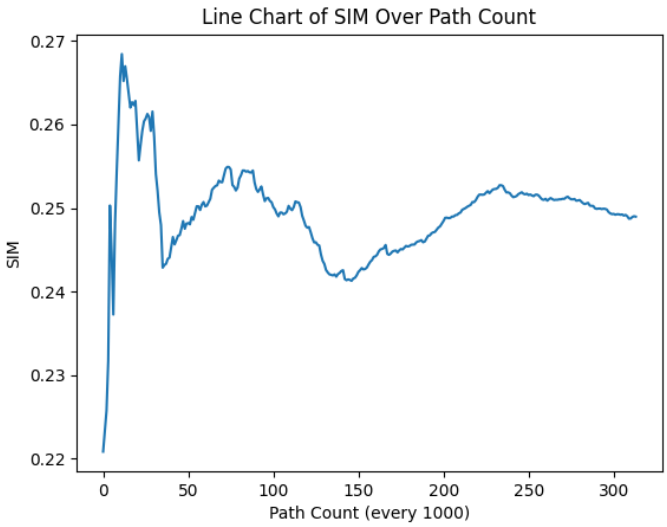


Figure 6: Line Chart of SIM Over Path Count

According to Figure 6, we can draw the conclusion that the value of SIM oscillates as the data is read in and eventually stabilizes around 0.25. And **the final value of SIM in Case 1 is 0.248971**.

Then, we use this model, which is capable of calculating SIM values, to compute the SIM value for the edge weight corresponding to Table 1 in the first task. The result is 0.75, which matches the result we obtained through iterative attempts. This indicates that our first task result is correct, and the model is indeed capable of effectively computing SIM values.

4.4 Model of Maximizing the SIM in Case 1

4.4.1 Simulated Annealing Algorithm

Simulated Annealing Algorithm is a global optimization algorithm commonly used to solve complex combinatorial optimization problems. The algorithm simulates the behavior of molecular motion during the annealing process of a solid, accepting solutions that are worse than the current solution with a certain probability to avoid getting trapped in local optima and to search for the global optimum.

The basic idea of the Simulated Annealing Algorithm originates from the physical phenomenon observed in the annealing process of solids. In the annealing process, by gradually lowering the temperature, atoms or molecules gain enough energy to escape from states with local energy minima, providing an opportunity to reach the global energy minimum state.

The Simulated Annealing Algorithm is typically used to compute the minimum value of an objective function. However, in our case, we need to find the maximum value of the objective function SIM. Therefore, some adjustments need to be made to the algorithm.

When making the logical decision of whether to accept a new SIM value, we modify the acceptance criterion. If the new SIM value is greater than the current SIM value, we accept the new SIM value with a probability of 100%. If the new SIM value is smaller than the current SIM value, the original algorithm would result in an exponential function with an exponent greater than 0, making the probability exceed

1. So in the expression, we add a negative sign to the exponent, as shown in the following formula:

$$P_{i+1} = \begin{cases} 1, SIM_{i+1} \geq SIM_i \\ \exp\left(\dfrac{SIM_{i+1} - SIM_i}{T_{i+1}}\right), SIM_{i+1} < SIM_i \end{cases}$$

P_{i+1} **represents the probability of accepting a new SIM value in the first round.** SIM_i **represents the SIM value in the i-th round, and** T_i **represents the temperature in the i-th round**, whose expression can be written as:

$$T_i = \lambda^i \cdot T_0$$

In the given code, T_0 **,**T_{end} **and** λ **are pre-set based on actual conditions**. According to many trials, we determine λ **as 0.9**. When T reaches T_{end}, the annealing process will automatically terminate, and the program stops running. The current SIM value represents the final result. The following Figure shows how the simulated annealing algorithm works:

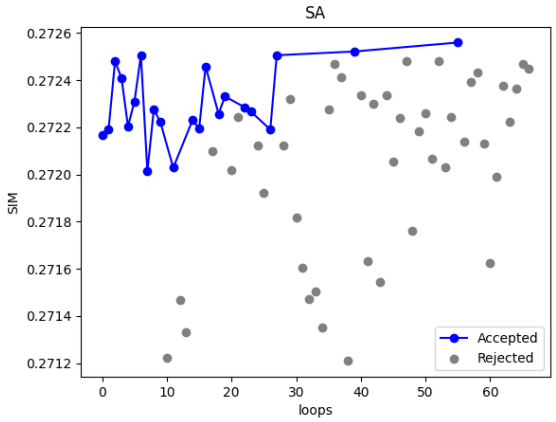


Figure 7: The curve of SIM value changes in An Annealing

The blue points in the graph represent the accepted results, indicating that the new SIM value is greater than the original SIM value. The gray points represent the rejected new SIM values, indicating that the new SIM value is smaller than the original SIM value and was not accepted. From Figure 7, we can observe that the SIM value exhibits a fluctuating upward trend during this annealing process. Additionally, there are several rejections between two consecutive acceptances in order to break out of a local optimum.

When calculating the objective function SIM, we need to compute the weights of each edge, so we need to determine the expression for edge weights. The expression for edge weights will be provided later. First, we will discuss one of the important factors in determining the weight expression, which is the importance of the roads. The next subsubsection will introduce the metric we adopted to measure the importance of roads, which is the PageRank of the edges.

4.4.2 PageRank and Edge PageRank

PageRank is an algorithm used to measure the importance or relevance of web pages within a network. It was developed by Google's co-founders, Larry Page and Sergey Brin. PageRank assigns

a numerical value called PageRank score to each web page in the network. The score is based on the idea that important web pages are likely to receive more links from other web pages. The PageRank score of a web page is influenced by the quantity and quality of incoming links. Web pages with higher PageRank scores are considered more authoritative and are given higher priority in search engine results. The PageRank formula in a general sense is described by the following formula:

$$PR(i) = (1-\gamma) + \gamma \cdot \sum_{1}^{n} \frac{PR(j)}{deg(j)}$$

$PR(i)$ represents the PageRank value of webpage i, $PR(1)$ to $PR(n)$ represent the PageRank values of other webpages that link to webpage i, $deg(1)$ to $deg(n)$ represent the outgoing link counts from other webpages to webpage i, and γ is a damping factor between 0 and 1 used to control the probability of random jumping.

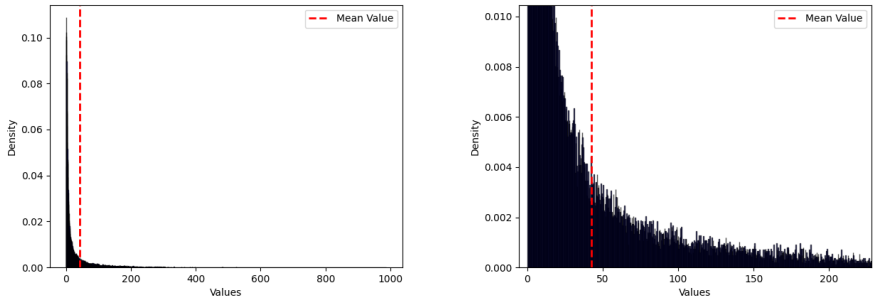
PageRank is mostly used for assessing the importance of webpages or entire websites, and it is rarely applied to evaluate the importance of edges. Therefore, we try to utilize this algorithm to measure the frequency of road usage.

For any arbitary graph G with vertex set V and edge set E, and for any edge e with u as the starting point and v as the endpoint, the edge PageRank value is defined as:

$$PR_e^{\gamma}(G) = \frac{1}{deg_u^+(G)}(\gamma \cdot \sum_{e' \in E_u^-(G)} PR_{e'}^{\gamma}(G) + \Phi(u))$$

$PR_e^{\gamma}(G)$ represents the PageRank value of edge e, $deg_u^+(G)$ represents the size of the set of all the outgoing edges, namely the out-degree of the vertex u, $\Phi(u)$ represents a empirical formula associated with u, and we give it **an initial value of 1** for convenience. **γ represents a parameter. Referring to previous literature, we assign a value of 0.85 to γ**. According to the recursive formula, as an edge is traversed more frequently, the sum of the edge PageRank values of its adjacent edges increases, resulting in a higher PageRank value for that edge.

We substituted the data for case 1 into the formula and performed the calculations. We then plotted the probability density graph of the edge PageRank values, as shown in the following figure:



(a) Probability Distribution Graph of All Edge PageRank

(b) Probability Distribution Graph of All Edge PageRank

Figure 8: Histogram of Edge PageRank Frequency Distribution

The left figure represents the overall distribution of edge PageRank values, while the right figure displays the data within the range of 0 to 250 for edge PageRank values. It can be observed that the majority of edges have PageRank values below 50. However, there are also some road edges with PageRank values exceeding 100 or even 200, indicating that using edge PageRank to determine the importance of roads exhibits significant differentiation and reliability.

4.4.3 The Straight-line Distance and Length of Roads

To incorporate the length of the road traveled, we consider both the road length and the straight-line distance between the starting point and the endpoint of the road when calculating the weight. This allows us to better capture potential user preferences for taking longer routes. Therefore, we define the length of road e as $length_e$ and the straight-line distance between the starting point and endpoint as $dist_e$.

The data for the road length has been provided in the question and processed accordingly. However, to calculate the straight-line distance of the road, we need to utilize the given longitude and latitude coordinates. For an edge e, given the latitude and longitude coordinates of its starting point u and endpoint v, the calculation formula of $dist_e$ is as follows:

$$dist_e = 1000 \times \sqrt{85.714 \times (lon_u - lon_v)^2 + 78.125 \times (lat_u - lat_v)^2}$$

In the formula above, lon_u **and** lon_v **represent the longitudes of the starting point and endpoint of road e, respectively**. lat_u **and** lat_v **represent the latitudes of the starting point and endpoint of road e respectively**. The straight-line distance between the two locations can be obtained by calculating the weighted sum of these values.

4.4.4 Initial Model

Now we can attempt to write the expression for the weight of an edge, denoted as ω:

$$\omega_e = \frac{\xi}{PR_e^{\gamma}(G)} + \zeta length_e + \eta dist_e$$

$PR_e^{\gamma}(G)$ **is the PageRank value of edge e,** $length_e$ **is the length of edge e, and** $dist_e$ **is the straight-line distance between the starting point and the endpoint of edge e**. ξ, ζ, and η are constants that need to be solved using the simulated annealing algorithm.

We utilize the simulated annealing algorithm to calculate three parameters that determine the weights, using the SIM value as the objective function, and record **the maximum SIM value is 0.273278**.

4.5 Model Optimizing

4.5.1 Data Re-mining

When we constructing the models, we also investigated the relationship between the PageRank value of each edge and the connection between its starting point and endpoint. The scatter plots are shown below in the respective figures:

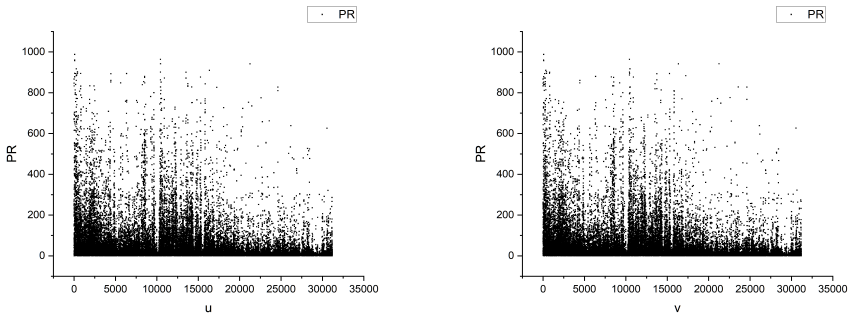


Figure 9: The Relationship between PR and u/v

From the two figures above, we can deduce there is **a significant "gap" in the PageRank values when the starting point or endpoint is around 10000**. Therefore, we believe it is necessary to investigate the relationship between the locations in this area and the notably lower PR values. As a result, we have marked the points from 9800 to 10200 on the map of Beijing, as shown in the figure below.

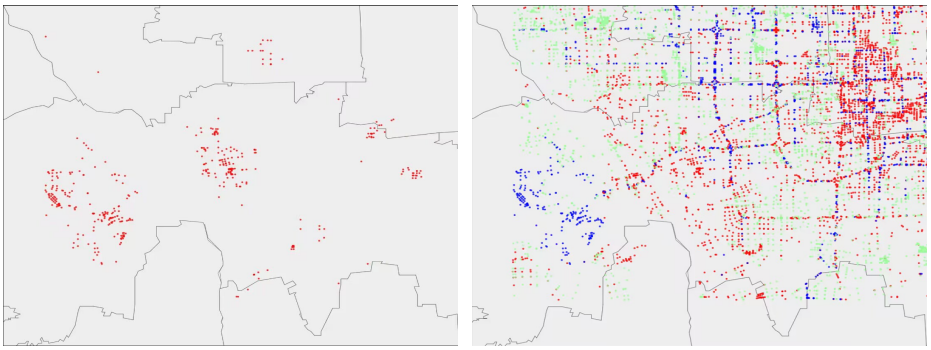


Figure 10: The Relationship between PR and u/v

From the figure, it can be observed that this subset of points is concentrated **on the southwest edge of the Beijing map**. Therefore, we believe that these points need to be treated differently and undergo special handling during the subsequent model optimization process.

At the same time, in Figure 9 the edge weights of points below 10000 are generally higher. The points ranging from 10000 to 20000 having high edge weights as well. So we proceed to mark points in different ranges with different colors and redraw the distribution of data points on the map. Points with sequence numbers less than 10000 will be shown in red, points with sequence numbers between 10000 and 20000 will be shown in blue, and points exceeding 20000 will be shown in green. The representation of all data points on the map of Beijing is shown with specific color in the figure below:

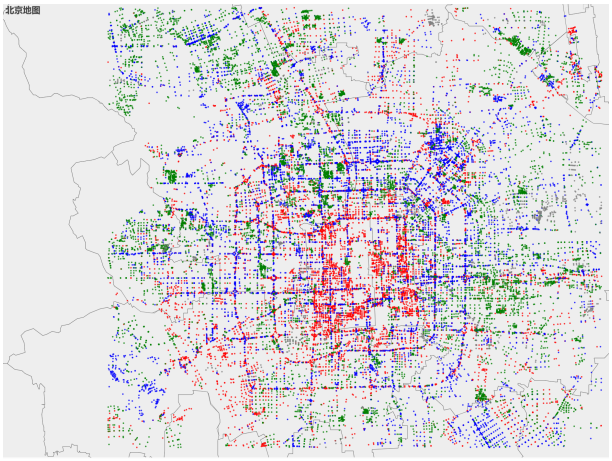


Figure 11: All the Vertices Shown on the Map

We discover a strong regularity in the distribution of the points: the red points are concentrated in the middle, while the blue points are primarily concentrated between the third and fourth ring roads. The distribution pattern of the green points are distributed on the outer side overall, which means points beyond 20000 do not exhibit a clear distribution pattern, but they are generally located in more remote areas, with lower corresponding edge pageranks.

Additionally, on the map, it is evident that there is a higher concentration of people traveling to important facilities or buildings such as subway lines, residential areas, and tourist attractions. To better illustrate the relationship between the distribution of the points and the railway transportation layout in Beijing, we have registered the previous map with the Beijing transportation network map. We have adjusted the transparency to enhance clarity, as shown in the following figure:

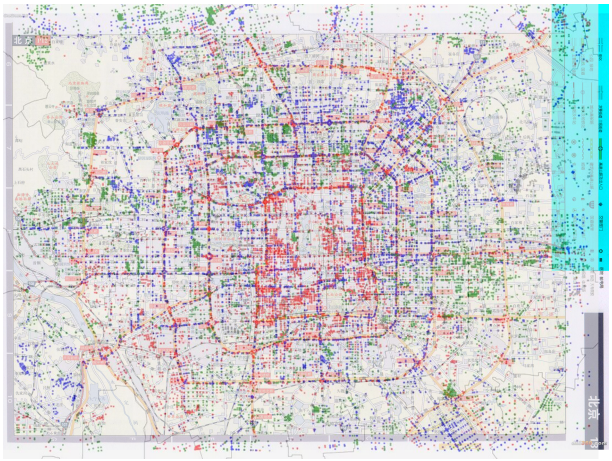


Figure 12: Vertices Shown on the Transportation Network of Beijing.

After the aforementioned data analysis, we have discovered a strong correlation between **the index of a point and its location**. Additionally, certain specific points represent special locations. Therefore, we believe that modifying $PR_e^{\gamma}(G)$ based on the index of a point can be beneficial.

At the same time, the third parameter η of the initial model seemes to **have a relatively minor impact on the SIM value**, as the following figure shows:

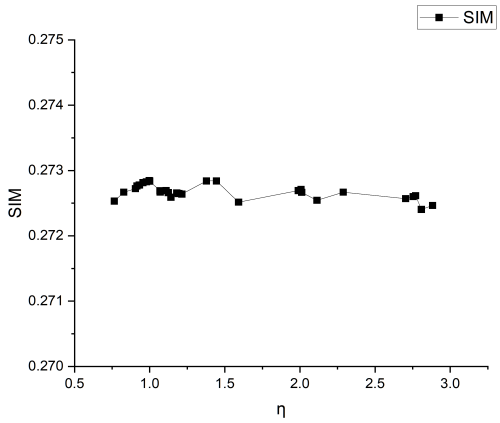


Figure 13: The Relationship between SIM and η

From the Figure, we can observe that when the two parameters remain constant, the impact of η on the model results is not significant. The latitude and longitude coordinates used in the $dist_e$ function can be represented by the index of the points. Therefore, we will remove the third parameter from the model and optimize the calculation method for $\Phi(u)$ of the $PR_e^\gamma(G)$.

4.5.2 Model Adjusting

By removing the third parameter and only keeping A and B, we obtain the following expression for the weight ω_e:

$$\omega_e = \frac{\xi}{PR_e^\gamma(G)} + \zeta length_e + C$$

The expression of $\Phi(u)$ in $PR_e^\gamma(G)$ is as shown in the following formula:

$$\phi(u) = \begin{cases} 2, 0 \leq u < 4800 \\ 1, 4800 \leq u < 9800 \\ 0.02, 9800 \leq u < 10200 \\ 1.4, 10200 \leq u < 16000 \\ 1.4e^{1.355-8.471\times10^{-5}u}, 16000 \leq u < 31198 \end{cases}$$

The expression in the last part is obtained by fitting a curve to the points in the right section of Figure 9.

The maximum SIM value achieved by our optimized model is 0.298223, while the maximum SIM value of the original model is 0.273278. **The optimization rate is calculated as 9.128%**.

4.6 Analysis and Comparison

From the calculations in the previous sections, it can be observed that the maximum SIM value obtained by considering the path length as the weight is 0.248971. The maximum SIM values obtained by the original and optimized models, which calculate the weights differently, are 0.273278 and 0.298223, respectively. The optimization rates are **9.76298% and 19.7822%**, respectively.

To better illustrate the difference between our method of calculating edge weights and the length of the edges, we have plotted a scatter plot showing the relationship between edge weights and lengths, as shown in the following figure:

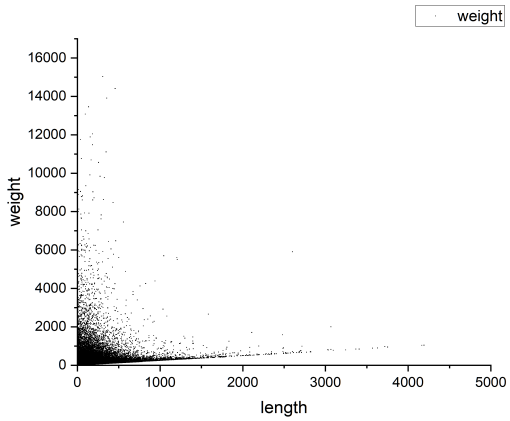


Figure 14: The relationship Between Weight And Length

From the Figure 14, the relationship between edge weights and edge lengths is not always linear. When the length is short, the weight varies significantly, indicating that people are less likely to choose some edges even if they are short in distance. As the edge length gradually increases, the weight generally tends to increase, and the weight becomes more stable with smaller variations. This indicates that most of the time people are unwilling to take detours. However, there are also edges with significantly lower weights when the length is larger. The presence of these edges is usually due to people wanting to avoid specific locations or needing to reach specific attractions.

To provide a more detailed explanation of the reasons for these differences, we compared the differences between edge PageRank and edge length, and plotted a scatter plot to illustrate this:

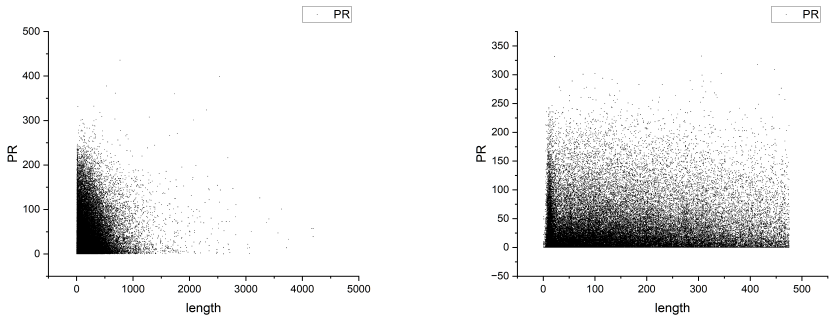


Figure 15: The relationship Between PR And Length

The left figure shows the relationship between the PageRank values (PR) of all points and their corresponding lengths (length). The right figure is a zoomed-in view of the left figure, focusing on the part where the length is less than 500. It is evident that when the length is small, there is a larger variation in PR values, whereas as the length increases, the PR values gradually stabilize.

By taking into account these two factors, we can effectively determine the weights of the roads. With the consideration of user experience, we have identified the optimal path for selection.

5 Testing

We have a model based on the simulated annealing algorithm, which is influenced by various parameters such as the maximum value of parameter variation (strides) and initial and final temperatures. We will conduct stability tests on these two parameters to evaluate the stability of the model.

First, we keep the initial and final temperatures and parameter values unchanged in the algorithm. We adjust the strides to be 0.3 and 0.6, respectively, and run the program. The final results are as follows:

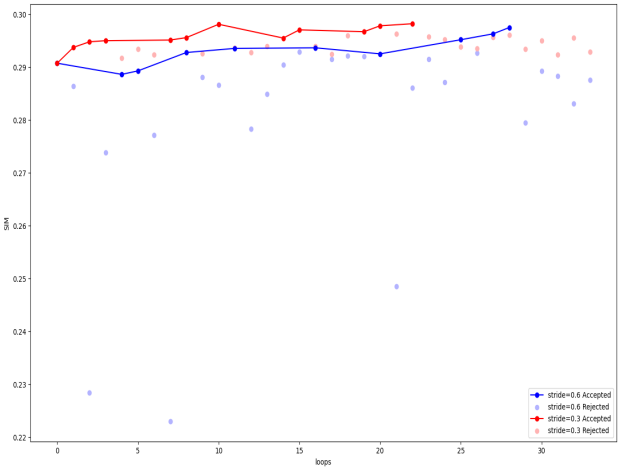


Figure 16: SIM with Different Strides

We can see that the group with a smaller step size reached the optimum value first and then remained relatively stable. On the other hand, the group with a larger step size took longer to reach the optimum value. The maximum SIM values obtained from the two groups with different step sizes are close, indicating that the model is not sensitive to this parameter.

Next, we kept the initial values and strides of all parameters unchanged and only adjusted the initial and final temperatures of the algorithm to be (50,1) and (25,0.5) with a purpose of controlling the rounds. The results are as follows:

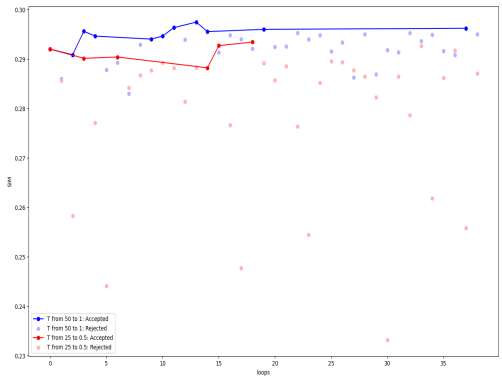


Figure 17: SIM with Different Initial and Final Temperatures

We found that the group with a higher initial temperature calculated relatively higher results, while the group with a lower temperature had a lower probability of accepting smaller values, making it easier to reject smaller values and resulting in relatively lower results. However, there was no significant difference between the two groups, indicating a stable performance.

To better showcase the tests we conducted, we have created the following table for comparison:

Table 2: Robustness Test

Trial	SIM_{max}
T_0=50 T_{end}=1 strides=0.3	0.296183
T_0=25 T_{end}=0.5 strides=0.3	0.293384
T_0=30 T_{end}=1 strides=0.6	0.297507
T_0=30 T_{end}=1 strides=0.3	0.298223

Considering the two indicators above, we believe that **our model exhibits strong stability and is not sensitive to changes in parameters**.

6 Strengths and Weaknesses

6.1 Strengths

- For task 2, we utilized the heap-optimized Dijkstra's algorithm to calculate the shortest paths. We categorized the paths based on the starting point and computed them, which allowed us to achieve extremely fast computational speed within limited space complexity.
- For task 3, we employed PageRank, which takes into account the relationships between nodes in terms of topology, geographic location, and the importance of edges. The model we used is computationally simple and highly scalable. Additionally, by utilizing a simulated annealing algorithm, we achieved fast convergence and a strong ability to escape local maxima.

6.2 Weaknesses

- Due to the limited amount of actual data, the generalizability of the model has not been thoroughly validated.
- Due to the difficulty of convergence in high-dimensional spaces for simulated annealing algorithms, our model structure is relatively simple and has relatively weak expressiveness. Strong computational support is required to improve performance in higher dimensions.
- Due to limited computational performance and time constraints, the number of iterations for simulated annealing was insufficient, and the solution is still slightly away from the optimal solution.

References

[1] Page L, Brin S, Motwani R, et al. The pagerank citation ranking: Bring order to the web[R]. Technical report, stanford University, 1998.

[2] Dijkstra E W. A note on two problems in connexion with graphs[M]//Edsger Wybe Dijkstra: His Life, Work, and Legacy. 2022: 287-290.

[3] Niu G, Cartoixaá X, Grossi A, et al. Mechanism of the Key Impact of Residual Carbon Content on the Reliability of Integrated Resistive Random Access Memory Arrays[J]. The Journal of Physical Chemistry C, 2017, 121(12): 7005-7014.

[4] Leitão A, Vinhas A, Machado P, et al. A Genetic Algorithms Approach for Inverse Shortest Path Length Problems[J]. International Journal of Natural Computing Research (IJNCR), 2014, 4(4): 36-54.

[5] Steinbrunn M ,Moerkotte G, Kemper A. Heuristic and Randomized Optimization for the Join Ordering Problem[J] . The VLDB Journal , 1997 , 6 (3) :8 - 17.

[6] Cui T, Hochbaum D S. Complexity of some inverse shortest path lengths problems[J]. Networks, 2010, 56(1): 20-29.

Appendix

```
inline void Dijkstra(int s, int* d, int numv, vector<Edge>* adj, unordered_map<pai
{
    priority_queue<pair<int, int>, vector<pair<int, int>>, greater<pair<int, int>>
    bool visited[maxV];
    for (int i = 0; i < numv; i++)
        d[i] = maxD;
    memset(visited, 0, sizeof(visited));
    d[s] = 0;
    int mind = maxD;
    int now = 0;
    q.push({ 0, s });
    while (true) {
        mind = maxD;
        if (q.empty())
            return;
        auto x = q.top();
        q.pop();
        int now = x.second, mind = x.first;
        if (visited[now])
            continue;
        visited[now] = 1;
        for (int i = 0; i < adj[now].size(); i++) {
            int to = adj[now][i].end;
```

```
                if (d[now] + map[{now, to}].weight < d[to])
                {
                    d[to] = d[now] + map[{now, to}].weight;
                    q.push({ d[to], to });
                }
            }
        }
    }

    static double SIM(unordered_map<pair<int, int>, Edge, pair_hash>& map, vector<Edge
ifstream file("path.txt");//data-processed
string line;
int simcnt = 0, cnt = 0, start0 = -1;
int d[maxV];
while (getline(file, line)) {
istringstream ss(line);

int pathlength = 0;
int start, end;
ss >> start;
if (start0 != start) {
dijkstra(start, d, numv, edges, map);
start0 = start;
}
while (ss >> end) {
pathlength += map[{start, end}].weight;
start = end;
}
if (pathlength == d[end]) {
simcnt++;
}
cnt++;
}

file.close();
return 1.0 * simcnt / cnt;
}


    static void Page_Rank(vector<Edge>* adj, vector<Edge>* antiadj, int numv, double*
vector <Edge>new_Adj[maxV];
vector <Edge>new_Antiadj[maxV];
for (int i = 0; i < numv; i++) {
double pr = 0;
for (int k = 0; k < antiadj[i].size(); k++)
```

```
pr += antiadj[i][k].weight;
pr *= a;
pr += b[i];
pr /= adj[i].size();
for (int j = 0; j < adj[i].size(); j++)
{
add_Edge(i, adj[i][j].end, pr, new_Adj);
add_Edge(adj[i][j].end, i, pr, new_Antiadj);
}
}
for (int i = 0; i < numv; i++)
{
for (int j = 0; j < adj[i].size(); j++)
adj[i][j] = new_Adj[i][j];
for (int j = 0; j < antiadj[i].size(); j++)
antiadj[i][j] = new_Antiadj[i][j];
}
}

    struct pa {
double a, b, c;
};
static pa SA(pa now, unordered_map<pair<int, int>, Edge, pair_hash>& pr, unordered_map
unordered_map<pair<int, int>, Edge, pair_hash> weight;
cal_weight(now, weight, pr, length, v);
double sim1, sim2, dsim;
sim1 = SIM(weight, edges);
double T = 25, T_end = 0.5;
while (T > T_end) {
pa par = update(now);
cal_weight(par, weight, pr, length, v);
sim2 = SIM(weight, edges);

dsim = (sim2 - sim1) * 10000;
double r;
r = (double)rand() / RAND_MAX;
if (r < exp(dsim / T)) {
now = par;
sim1 = sim2;
}
}
else {
now = par;
sim1 = sim2;
}
```

```
T *= 0.9;
}
myfile.close();
return now;
}
```