Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

APFD: an effective approach to taxi route recommendation with mobile trajectory big data*

Wenyong ZHANG[†1], Dawen XIA[†‡1], Guoyan CHANG[5], Yang HU[2], Yujia HUO[1], Fujian FENG[1], Yantao LI[3], Huaqing LI[†‡4]

[1]*College of Data Science and Information Engineering, Guizhou Minzu University, Guiyang 550025, China*

[2]*Department of Automotive Engineering, Guizhou Traffic Technician and Transportation College, Guiyang 550008, China*

[3]*College of Computer Science, Chongqing University, Chongqing 400044, China*

[4]*College of Electronic and Information Engineering, Southwest University, Chongqing 400715, China*

[5]*The Affiliated Hospital of Guizhou Medical University, Guiyang 550001, China*

[†]E-mail: gzmdzwy@gzmu.edu.cn; dwxia@gzmu.edu.cn; huaqingli@swu.edu.cn

Received Nov. 14, 2021; Revision accepted June 24, 2022; Crosschecked Aug. 1, 2022; Published online Sept. 24, 2022

Abstract: With the rapid development of data-driven intelligent transportation systems, an efficient route recommendation method for taxis has become a hot topic in smart cities. We present an effective taxi route recommendation approach (called APFD) based on the artificial potential field (APF) method and Dijkstra method with mobile trajectory big data. Specifically, to improve the efficiency of route recommendation, we propose a region extraction method that searches for a region including the optimal route through the origin and destination coordinates. Then, based on the APF method, we put forward an effective approach for removing redundant nodes. Finally, we employ the Dijkstra method to determine the optimal route recommendation. In particular, the APFD approach is applied to a simulation map and the real-world road network on the Fourth Ring Road in Beijing. On the map, we randomly select 20 pairs of origin and destination coordinates and use APFD with the ant colony (AC) algorithm, greedy algorithm (A*), APF, rapid-exploration random tree (RRT), non-dominated sorting genetic algorithm-II (NSGA-II), particle swarm optimization (PSO), and Dijkstra for the shortest route recommendation. Compared with AC, A*, APF, RRT, NSGA-II, and PSO, concerning shortest route planning, APFD improves route planning capability by 1.45%–39.56%, 4.64%–54.75%, 8.59%–37.25%, 5.06%–45.34%, 0.94%–20.40%, and 2.43%–38.31%, respectively. Compared with Dijkstra, the performance of APFD is improved by 1.03–27.75 times in terms of the execution efficiency. In addition, in the real-world road network, on the Fourth Ring Road in Beijing, the ability of APFD to recommend the shortest route is better than those of AC, A*, APF, RRT, NSGA-II, and PSO, and the execution efficiency of APFD is higher than that of the Dijkstra method.

Key words: Big data analytics; Region extraction; Artificial potential field; Dijkstra; Route recommendation; GPS trajectories of taxis

https://doi.org/10.1631/FITEE.2100530 **CLC number:** TP39

1 Introduction

Along with social progress, people's capacity for consumption and travel is increasing, which creates

‡ Corresponding authors

* Project supported by the National Natural Science Foundation of China (Nos. 62162012, 62173278, and 62072061), the Science and Technology Support Program of Guizhou Province, China (No. QKHZC2021YB531), the Youth Science and Technology Talents Development Project of Colleges and Universities in Guizhou Province, China (No. QJHKY2022175), the Science and Technology Foundation of Guizhou Province, China (Nos. QKHJCZK2022YB195 and QKHJCZK2022YB197), the Natural Science Research Project of the Department of Education of Guizhou Province, China (No. QJJ2022015), and the Scientific Research Platform Project of Guizhou Minzu University, China (No. GZMUSYS[2021]04)

ORCID: Wenyong ZHANG, https://orcid.org/0000-0002-6969-9695; Dawen XIA, https://orcid.org/0000-0002-0151-9643; Huaqing LI, https://orcid.org/0000-0001-6310-8965

© Zhejiang University Press 2022

higher requirements for urban transportation. Recently, taxi has become the first choice for travelers due to its convenience, safety, and personalization (He et al., 2018; Lai et al., 2019). Specifically, it is well known that urbanization has led to the growth of traffic congestion. To address this issue, urban managers have invested significant funds to expand and renovate urban roads, making the transportation network increasingly complicated. At the same time, the complex road network and traffic congestion have made taxis less efficient in carrying passengers (Qin et al., 2017). Existing studies have shown that taxis spend 35%–60% of their time cruising the road and searching for passengers (Schaller Consulting, 2006). On one hand, this increases waiting time and creates a poor ride experience for passengers. On the other hand, it results in issues such as high taxi operating cost, high energy consumption, and environmental pollution (Ji et al., 2020). To improve passenger experience, increase the efficiency of taxi operation, and reduce energy consumption, many scholars have proposed different methods for recommending the fastest taxi routes. Currently, route planning methods can be divided into classic methods and heuristic methods (Mac et al., 2016; Zajac and Huber, 2021). The dynamic programming method, branch-and-bound method (Przybylski and Gandibleux, 2017), artificial potential field (APF) method, Dijkstra method, Bellman–Ford method (Parimala et al., 2021), and greedy algorithm (A*) (Zhang Y et al., 2019) are classic methods. The latter includes simulated annealing (Lin et al., 2021), firefly algorithm (Zhang XJ et al., 2019), neural network method (Wang JK et al., 2020), genetic algorithm (Nazarahari et al., 2019), ant colony (AC) algorithm (Jiao et al., 2018), and Dolphin swarm algorithm (Wu TQ et al., 2016).

When it comes to heuristic algorithms, the simulated annealing method has strong local search ability, and the neural network method easily converges to the local minimum solution when the target scale is large. The genetic algorithm, with excellent robustness, has strong parallel processing ability and global search ability. However, it has poor local search ability, thus resulting in premature convergence. The characteristics of the AC algorithm are self-adaptation, self-organization, and positive feedback falling, which can promote the whole system to evolve to the optimal solution. However, the AC algorithm has the shortcomings of low convergence speed and is easy to fall into local optima. In addition, there is no parameter setting in the algorithm, and the theoretical basis must be adjusted and determined by extensive experiments (Agarwal and Bharti, 2020; Sharma and Doriya, 2020).

Among classic approaches, dynamic programming is a recursive method, and the time complexity grows sharply as the number of targets increases. The time complexity of the branch-and-bound method increases with an increase in the number of targets, and the calculation time is longer when the number of targets is larger. The APF method imitates the movement of objects under gravity and repulsion. Gravitational forces exist between the target point and moving object. There are repulsive forces between the moving object and obstacles. The route is optimized by establishing a gravitational field function and a repulsive force field function. The advantage is that the route planning effect is smooth, but many problems exist, such as being easily trapped into local optima. Dijkstra is a classic method used to search for the shortest route, and it extends from the origin to the outer layer to the end. The shortest route is obtained by comparing all nodes because the method needs to gradually expand from the origin to the surroundings. Therefore, the calculation of the shortest route has a high success rate and good robustness, but when it is applied to a large-scale node network, the execution time is longer than that of other algorithms (Hoy et al., 2015; Mavrovouniotis et al., 2017; Rizk et al., 2018; Zajac and Huber, 2021).

When taxis carry passengers, the method needs to complete the optimal route recommendation in the shortest possible time. If the execution time of the method is too long, taxi drivers and passengers would wait for a long time for the route planning results. If the method fails to obtain the optimal route, it affects the arrival time of passengers. Therefore, to satisfy passengers and taxi drivers, the recommendation result of the algorithm needs to have the shortest execution time and the best route planning.

Heuristic methods have strong search capabilities and large-scale retrieval capabilities compared to classic methods, but they require multiple iterations to obtain a better route. Although classic methods take a long time to process a large-scale node network, when the number of nodes is smaller

than a certain size, the execution efficiency is higher. In classic methods, Dijkstra finds the destination by traversing layer by layer, and the route obtained is the optimal route. In the calculation process, most of the points cannot be the optimal route nodes. If they can be removed, the execution time of the Dijkstra method would be greatly reduced.

For this reason, the algorithm needs to be executed in a short time and solve the recommended route optimization problem. We propose a novel hybrid route recommendation approach (named APFD) based on the APF and Dijkstra methods. Specifically, according to the origin and destination coordinates, we select the area that contains the best route. Furthermore, we employ the APF method to remove redundant points with a low possibility of forming the shortest route. Finally, we use the Dijkstra method for optimal route planning.

The proposed APFD approach is applied to the simulated road network and the real-world road network on the Fourth Ring Road in Beijing. Compared with AC, A*, APF, rapid-exploration random tree (RRT), non-dominated sorting genetic algorithm-II (NSGA-II), and particle swarm optimization (PSO) methods, route planning ability of APFD is the best. Particularly, compared with Dijkstra, APFD has higher execution efficiency.

The main contributions of this work are summarized as follows: (1) The origin and destination coordinates are used to extract the area that is likely to include the optimal route. The area is taken as the effective area for method retrieval to improve the execution efficiency. (2) A potential energy field is constructed in the road network area. A critical equipotential line is established according to the current coordinate position, and redundant points are removed by comparing the potential energy of each connection point with the critical potential. (3) A distance matrix, a marker matrix, and a visited matrix are constructed for calculating the optimal route. These matrices are calculated using the node coordinates and side length data to obtain the optimal route.

2 Related works

In this section, we review related works about route recommendation and planning.

The rapid development of information technology to solve problems caused by urban transportation has led to the intelligent transportation system. Optimal route recommendation is an important branch and key technology in the field of intelligent transportation (Wu N et al., 2019; Yang et al., 2021). In terms of massive and rapid changes in traffic information, it is very important that the intelligent transportation system can complete route planning before the traffic information is updated and efficiently transferred to drivers (Algfoor et al., 2017; Santos et al., 2018; Wang YS et al., 2021). Therefore, the performance of the route planning method becomes the key to solve this problem. For this reason, many scholars have conducted research on route planning methods.

Using heuristic methods, Nimmagadda et al. (2020) proposed an adaptive planning algorithm that uses a combination of continuous space and spatial discretization methods to generate optimal paths and achieve fast route convergence. The algorithm is highly efficient and robust. Ajeil et al. (2020) put forward a hybrid algorithm combining a PSO algorithm and an improved frequency bat algorithm to solve the route planning problem. This method can generate the best feasible route, even in a complex dynamic environment, and thus overcomes the shortcomings of traditional methods such as the grid method. In addition, compared with existing route planning technologies, the proposed hybrid particle swarm optimization-modified frequency bat (PSO-MFB) algorithm is strongly competitive in route optimization. Wang JK and Meng (2020) presented a non-uniform sampling technique based on rapid exploration of random tree pipelines, which was used to calculate high-quality conflict-free paths while maintaining fast asymptotic convergence to the optimal solution. The algorithm first calculates the heuristic route. Furthermore, heuristic route discretization and multiple potential functions provide non-uniform sampling distribution for the route planner. Compared to advanced route planning algorithms, this algorithm has better performance.

Although the above methods have strong search capabilities and large-scale retrieval capabilities, they require multiple iterations to obtain a better route. Also, the execution time of the algorithms is long, and it is difficult to meet the real-time requirements.

For classic methods, Sinyukov and Padir (2020)

designed a high-performance algorithm based on two-dimensional grid single-source route planning at any angle. This algorithm abandons the general graph model, and directly uses discrete geometric primitives to represent vertices on the grid geometry. By introducing floating-point calculations, the accuracy was enhanced, and a better optimal route was obtained. Danilovic et al. (2021) developed a breadth-first search method to solve the shortest route problem, and all points were processed hierarchically. The method starts from the source point and scans to the outer layer, step by step, to select the most suitable point in the next layer until the key layer is found. Huang et al. (2021) proposed a multi-weight and multi-destination route planning framework with deadline constraints. The experimental results showed that the proposed algorithm performed well in terms of efficiency and effectiveness. Qu et al. (2020) proposed an adaptive shortest expected cruise route method using taxi global position system (GPS) data, the Kalman filter method, a probabilistic network model, and the data structure KDS-tree to improve recommendation efficiency. Zhu and Sun (2021) put forward the reverse labeling Dijkstra algorithm (RLDA) to solve the problem of car rental route recommendation based on the traditional Dijkstra algorithm, breadth-first search, and data structure of stack and queue. The experimental results demonstrated that this method was better than the PSO algorithm, genetic algorithm, AC algorithm, and neural network algorithm, and particularly had higher convergence efficiency and higher calculation speed. Akram et al. (2021) designed a Dijkstra algorithm for the shortest route of image blur, defining the trapezoidal image blur number and its graphical representation. The related cost was captured by the trapezoidal image blur number. Kou et al. (2020) developed a dichotomous method based on the Lagrangian dual method, employing the Dijkstra method in the Lagrangian dual method to improve the efficiency of the algorithm execution, and the effect was appreciable when addressing the route recommendation problem for large-scale complex networks.

Some of these studies adopted the idea of improving the classic route planning method to achieve optimal route planning. These algorithms are highly efficient, but it is difficult to obtain the optimal route using these algorithms. Other research focused on the use of hybrid algorithms for optimal route planning. The route planning ability is strong, but the execution efficiency is low, and it is difficult to meet the requirements of fast route planning. In addition, although some studies mentioned above employed the Dijkstra optimization method for route planning, their focus was on solving the route recommendation problem for large-scale complex networks, and the effect was not necessarily optimal based on the number of small- and medium-scale nodes. The APFD method proposed in this paper reduces the influence of redundant points on the execution efficiency of the algorithm by extracting effective areas and using APFs. After removing redundant points, the Dijkstra algorithm is used to calculate the optimal route plan, which guarantees the performance of the algorithm. The execution time is short, and the optimal route is guaranteed.

3 Problem analysis

In this section, we describe the taxi route recommendation problem and then put forward some reasonable hypotheses. Finally, the problem is modeled mathematically.

3.1 Problem statement

The road network can be expressed and constructed in various ways. In this work, we define intersections in the road network as road network nodes, roads between intersections as road side lengths, and the actual length of road side lengths as the weight value. Fig. 1 shows a route simulation. The red node represents the road network node, the black line segment is the length of the road, point D represents the destination location of the passenger, and point O denotes the origin position of the taxi. At the current location, the taxi is required to reach the passenger location quickly after receiving a passenger's request.

3.2 Problem hypothesis

According to the problem statement, we give the following assumptions:

1. Road network nodes and road side lengths are undirected graph networks whose attributes do not change over time. A taxi on each road can exit from both ends of the node.

2. The distance from turning left, turning right, going straight, or making a U-turn can be negligible.

3. When a taxi departs from any position on the road, the origin is defined as the road network node that is closest to this point. As illustrated in Fig. 2, points A and B are road network nodes, and point C is the passenger location point or taxi departure point, L_1 is the actual distance from C to A, and L_2 is the actual distance from C to B. If $L_1 > L_2$, the passenger or taxi position is A.

3.3 Problem definition

According to the problem statement and assumptions, the mathematical model is defined as follows:

We define a ternary undirected graph $G(V, E, W)$. V is the intersection node set, $V = \{v_1, v_2, \cdots, v_n\}$, and $|V|$ is the number of intersection nodes ($|V| = n$). E is the edge set between any two intersection nodes, and $|E|$ is the number of road sides ($|E| = t$). W is the weight value of E (the weight is measured by the value of the distance), and $W = \{w_1, w_2, \cdots, w_t\}$.

Suppose that $O = S_0$ is the origin node and that $D = S_k$ is the destination node, from O to D. All network nodes that the taxi passes are $S_0, S_1, \cdots, S_k$. Let $w_i(S_i, S_{i+1})$ be the distance between two adjacent nodes, and we define $f(S)$ as the objective function of the total distance of the taxi from point O to point D. Then we have

$$f(S) = \sum_{i=0}^{k-1} w_i\left(S_i, S_{i+1}\right). \tag{1}$$

Therefore, the shortest route $L_{\min}$ is defined as the shortest distance:

$$L_{\min} = \min f(S) = \min \sum_{i=0}^{k-1} w_i\left(S_i, S_{i+1}\right). \tag{2}$$

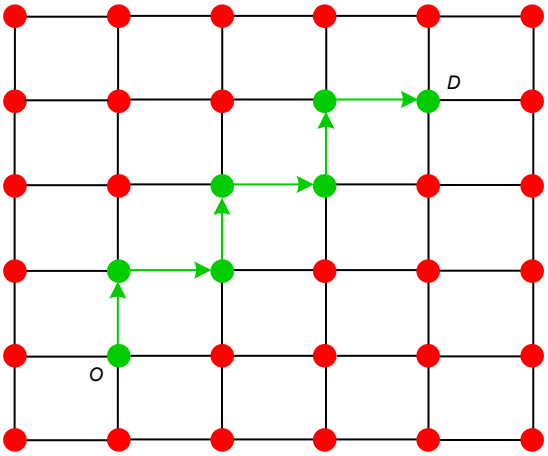

Fig. 1 Route simulation (References to color refer to the online version of this figure)

Fig. 2 Selection of the origin or destination nodes

4 APFD method

In this section, we describe the proposed APFD method in detail, including area extraction, redundant node removing, and route recommendation.

The process of the APFD method is shown in Fig. 3. Fig. 3a is the road network map, O is the origin point, and D is the destination point. First of all, the coordinates of O and D are employed to extract the area that is likely to have the optimal route and also is the effective area for method retrieval, as illustrated in Fig. 3b. Then, a potential energy field is constructed in the road network area, a critical equipotential line is established according to the current coordinate position, and redundant points are removed by comparing the potential energy of each connection point with the critical potential. According to the above two steps, the efficiency of the APFD method can be improved. Finally, a distance matrix, a marker matrix, and a visited matrix are constructed, as depicted in Fig. 3c. Also, three matrices mentioned above are calculated according to the node coordinates and side length data to obtain the optimal route, as plotted in Fig. 3d.

4.1 Effective area extraction

We coordinate all the road network nodes in Fig. 3a. The coordinates of origin O are (x_o, y_o), and the coordinates of D are (x_d, x_d). Ω is a matrix region coordinate set with O and D as the diagonal points. According to the coordinates of points O and D, the maximum x-coordinate, the minimum x-coordinate, the maximum y-coordinate, and the minimum y-coordinate in the Ω area can be derived:

$$\begin{cases} x_{\max} = \max\left(x_o, x_d\right), \\ x_{\min} = \min\left(x_o, x_d\right), \end{cases} \tag{3}$$

$$\begin{cases} y_{\max} = \max\left(y_o, y_d\right), \\ y_{\min} = \min\left(y_o, y_d\right). \end{cases} \tag{4}$$

According to Eqs. (3) and (4), the coordinates of the four vertices of the Ω region can be calculated as

$$(x_{\min}, y_{\min}), (x_{\min}, y_{\max}), (x_{\max}, y_{\min}), (x_{\max}, y_{\max}). \tag{5}$$

According to expression (5), all coordinate points in the Ω area can be extracted, and the extraction method is described in Algorithm 1.

4.2 Redundant node removing

In this subsection, we introduce the method of identifying and removing redundant nodes in detail. When the algorithm selects the next node from the current node, some nodes may not form the shortest route, which are called redundant nodes. If the redundant nodes can be removed before the algorithm is executed, the efficiency of the algorithm would be effectively improved. As shown in Fig. 4, point D is the destination, $P_{\rm s}$ is the current point, and $P_{\rm h}$ is the former point. $P_1, P_2, \cdots, P_f$ are the points that can be reached by $P_{\rm s}$, and $P_{\rm h}$ is not the arrival point selected by $P_{\rm s}$. Some of the nodes in $P_1, P_2, \cdots, P_f$ are redundant.

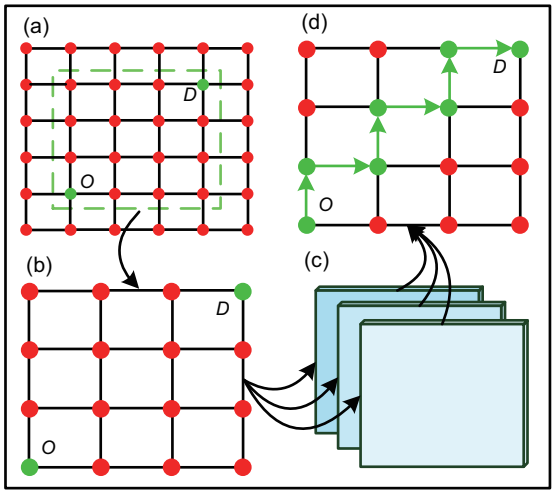

Fig. 3 Process of the APFD method: (a) simulating the road network; (b) extracting the effective area; (c) establishing the calculation matrices; (d) searching for the optimal route

Algorithm 1 Region extraction method

Require: pixel coordinates $O(x_o, y_o)$ and $D(x_d, y_d)$
1: $n_1 \leftarrow |x_o - x_d|, m_1 \leftarrow |y_o - y_d|$
2: $x_{\min} \leftarrow \min\left(x_o, x_d\right), y_{\min} \leftarrow \min\left(y_o, y_d\right)$
3: **for** $x_{\min} + i$ $(i \leftarrow 0, 1, \cdots, n_1 - 1)$ **do**
4: **for** $y_{\min} + j$ $(j \leftarrow 0, 1, \cdots, m_1 - 1)$ **do**
5: $\Omega \leftarrow (x_{\min} + i, y_{\min} + j)$
6: **end for**
7: **end for**
Ensure: Ω

To remove redundant points, we define the potential energy field ϕ with destination D as the center in the Ω region. The potential energy function is defined as

$$U_{\rm att}(q) = \frac{1}{2}\xi\rho^2\left(q, q_{\rm goal}\right), \tag{6}$$

where ξ is the positive proportional gain coefficient in the potential field, $\rho(q, q_{\rm goal})$ represents the absolute value of the distance between the current location of the taxi and the targeted position, $\rho = |q - q_{\rm goal}|$, q is the current position, and $q_{\rm goal}$ is the targeted position.

According to Eq. (6), the potential energy distribution in the Ω region is plotted in Fig. 5. We take point D as the center, and the potential energy gradually increases outward along point D. The potential energy field ϕ is continuously distributed in the Ω region, and the dashed circle is the equipotential. Same potential energy is on the same equipotential line, and the potential energy changing from point O to point D is shown in Fig. 6.

As illustrated in Fig. 6, point O is the farthest away point from point D and has the largest potential energy value. The potential energy decreases continuously from point O to point D. Point D has the smallest potential energy value. Moving from point O to point D can be regarded as an edge movement in a direction along which the gradient gradually decreases. We define the attractive force

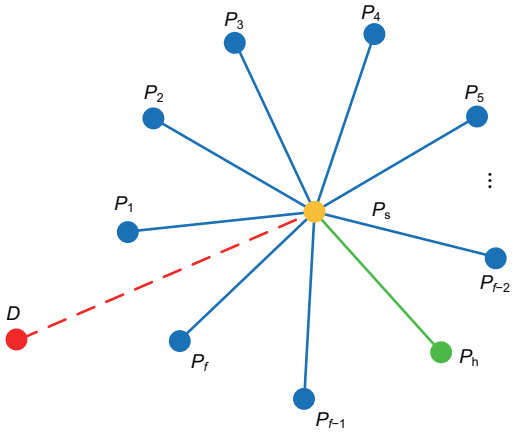

Fig. 4 Point connection with any point

of point D to any point in the potential energy as the negative gradient of the attractive potential energy:

$$-\nabla U_{\mathrm{att}}(q)=\xi\left(q_{\mathrm{goal}}-q\right). \tag{7}$$

As illustrated in Fig. 5, we take point P_3 as an example. An APF method is used to remove redundant points connected to point P_3. Point P_3 is set as the current position point. Meanwhile, the points connected to it are P_1, P_2, P_4, and P_5. According to the gravitational Eq. (6), the potential energy relations of P_1, P_2, P_4, P_5, and P_3 are

$$E_{P_1}>E_{P_2}>E_{P_3}>E_{P_4}>E_{P_5}. \tag{8}$$

We define the equipotential line where the current position point P_3 is located as the critical equipotential line. As shown in Fig. 7, the red equipotential line where P_3 is located is the critical equipotential line. We expect to get closer to point D at every step. Therefore, the next step is to move in the negative gradient direction:

$$\Delta U=E_q-E_{q_{\mathrm{pre}}}<0, \tag{9}$$

where E_q denotes the potential energy value of the current position, and $E_{q_{\mathrm{pre}}}$ represents the potential energy value of the approaching point.

Therefore, we judge the points whose potential energy values are greater than the critical equipotential as redundant points in Fig. 7. Points P_1 and P_2 are redundant points. Redundant points are directly marked as visited points during route calculation, thereby reducing calculation time and improving execution efficiency.

In special cases, as depicted in Fig. 4, if the potential energy at the P_{s} point is less than that of each point $P_1,P_2,\cdots,P_f$, or the potential energy at P_{s} is greater than the potential energy of each point $P_1,P_2,\cdots,P_f$, there are no redundant nodes. During route calculation, the route of each point $P_1,P_2,\cdots,P_f$ is completely calculated, and the process of removing redundant nodes is described in Algorithm 2.

4.3 Route recommendation

As shown in Fig. 8, according to the size of the coordinate point matrix of the Ω area, a marker matrix (Fig. 8a), a distance matrix (Fig. 8b), and a visited matrix (Fig. 8c) are established in this work.

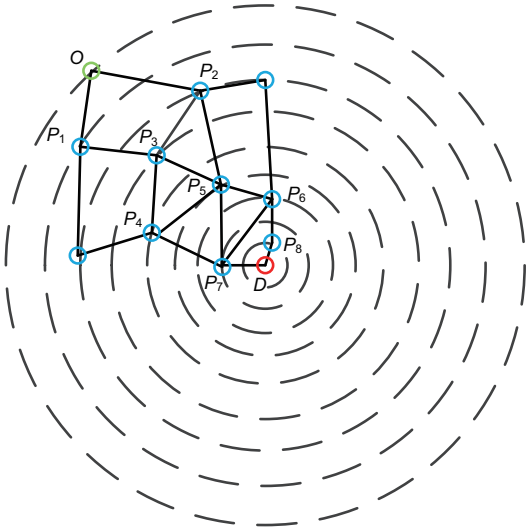

Fig. 5 Potential energy definition (Ω)

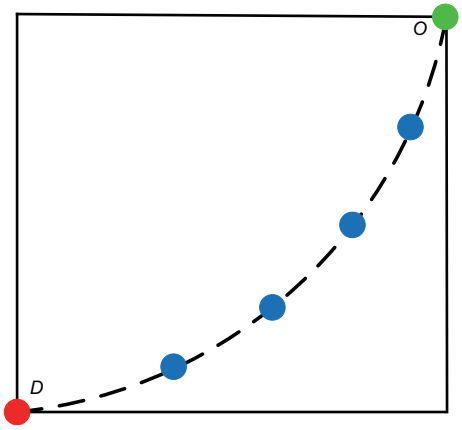

Fig. 6 Potential energy gradient direction

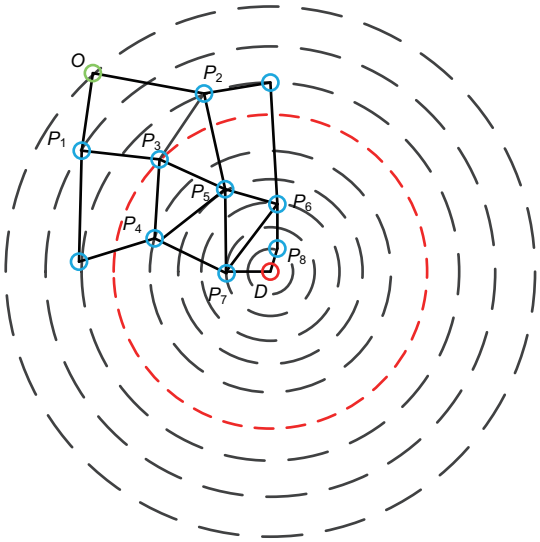

Fig. 7 Critical equipotential line for distinguishing redundant points (References to color refer to the online version of this figure)

Fig. 8a describes the marker matrix, which is used to mark whether the node has been visited. If the node has been visited, the mark is 1; otherwise, it is 0. Fig. 8b represents the distance matrix, which is used to record the minimum distance value from the origin to the current point. At the beginning of the algorithm, the distance value of point O is 0, and the other points are infinite. During the route calculation, the distance value of each point is continuously updated until the end of the calculation. Fig. 8c illustrates the visited matrix, which is used to record the coordinates of a point before the current position. The process of the algorithm is described as follows:

Step 1: initialize the matrices

We initialize the marker matrix, distance matrix, and visited matrix in Eqs. (10)–(12):

$$a(v)=0,\quad v\in V_1, \tag{10}$$

$$b(v)=\begin{cases}0, & v=O,\\ \infty, & v\neq O,\end{cases} \tag{11}$$

Algorithm 2 Removing redundant nodes

Require: pixel coordinates $D(x_d,y_d)$ and $P_{\rm s}(x_{P_{\rm s}},y_{P_{\rm s}})$
Read the nodes connected to PF $\leftarrow\{P_1,P_2,\dots,P_f\}$
$\rho_{P_{\rm s}}\leftarrow|P_{\rm s}-D|,\xi\leftarrow1$
$E_{P_{\rm s}}\leftarrow\frac{1}{2}\xi\rho_{P_{\rm s}}^2$
for $i\leftarrow1,2,\cdots,f$ **do**
  $\rho_{P_i}\leftarrow|P_i-D|$
  $E_{P_i}\leftarrow\frac{1}{2}\xi\rho_i^2$
  if $E_{Pi}>E_{P_{\rm s}}$ **then**
    Delete node P_i (P_i is a redundant node)
  end if
end for
Ensure: PM $=\{P_1,P_2,\dots,P_m\}$

$$c(v)=\begin{cases}O, & v=O,\\ \varnothing, & v\neq O,\end{cases} \tag{12}$$

where O is the origin and V_1 is the set of all the nodes in Ω.

In Eq. (12), when the targeted point is the initial position point, the current position is saved at the initial position point of the visited matrix, and the other positions are saved as empty.

Step 2: find the current point

The distance matrix is traversed to find the coordinate point with the smallest distance. In the marker matrix, we check whether the point has been marked. If it has been marked, we search again until the coordinate point of unmarked minimum distance is found, and then the point is regarded as the current point $P_{\rm s}$.

Step 3: remove redundant points

The set of coordinate points connected to point $P_{\rm s}$, PF $=\{P_1,P_2,\cdots,P_f\}$, is extracted with Eq. (6). We calculate the potential energy $E_{P_{\rm s}}$, E_{P_1}, E_{P_2}, $\cdots$, E_{P_f}. Comparing E_{P_1}, E_{P_2}, $\cdots$, E_{P_f} with $E_{P_{\rm s}}$, the coordinates with the potential energy greater than $E_{P_{\rm s}}$ are redundant points, which are marked as 1 in the corresponding coordinates of the marker matrix. We extract the coordinate point set as PM $=\{P_1,P_2,\cdots,P_m\}$ with the potential energy of the coordinate points being less than $E_{P_{\rm s}}$.

Step 4: update allowed access point data

The distances between all the nodes in the PM set and the $P_{\rm s}$ point are calculated, and let these be $L_1=\{l_1^1,l_2^1,...,l_m^1\}$. In the distance matrix, let the distance value at the coordinates of the $P_{\rm s}$ point be

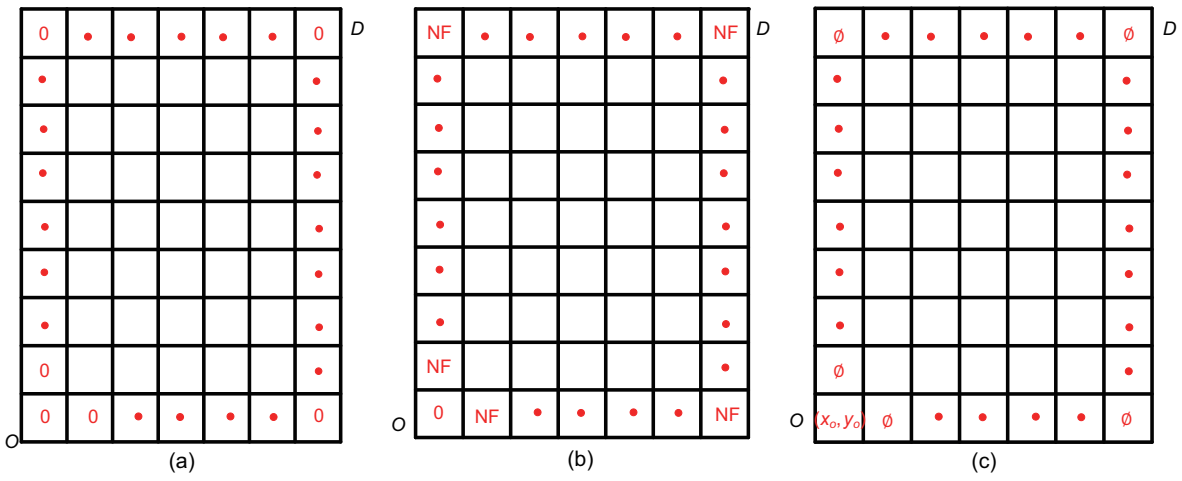

Fig. 8 Data matrices: (a) marker matrix; (b) distance matrix; (c) visited matrix

L and the distance value of each coordinate point in PM be L_3. We add L with all the distance values in L_1 to obtain the new distance values of coordinates of each point in PM, denoted as $L_2 = \{l_1^2, l_2^2, ..., l_m^2\}$. The distance values of each point in PM's current position are $L_3 = \{l_1^3, l_2^3, \cdots, l_m^3\}$. At this time, compared with each distance value in L_2 and L_3, if $l_i^2 < l_i^3$, we replace l_i^3 with l_i^2 in L_3 until all elements are compared. At this time, in the marker matrix, the P_s coordinate position is marked as 1, and in the visited matrix, the P_s coordinate values are stored at each coordinate position in the PM.

Step 5: calculate the route

Steps 2–4 are repeated until the destination coordinate D appears in PM, and then the calculation is stopped.

Step 6: extract the shortest route

We extract the shortest route by reverse derivation. In the visited matrix, the coordinates of the previous connection point D_{k-1} are extracted from the destination D coordinates, and we extract the coordinates of D_{k-2} at the coordinates of D_{k-1} until the coordinates of origin O are extracted. To this end, the route extraction is completed. The flowchart of APFD is shown in Fig. 9, and the execution process is described in Algorithm 3.

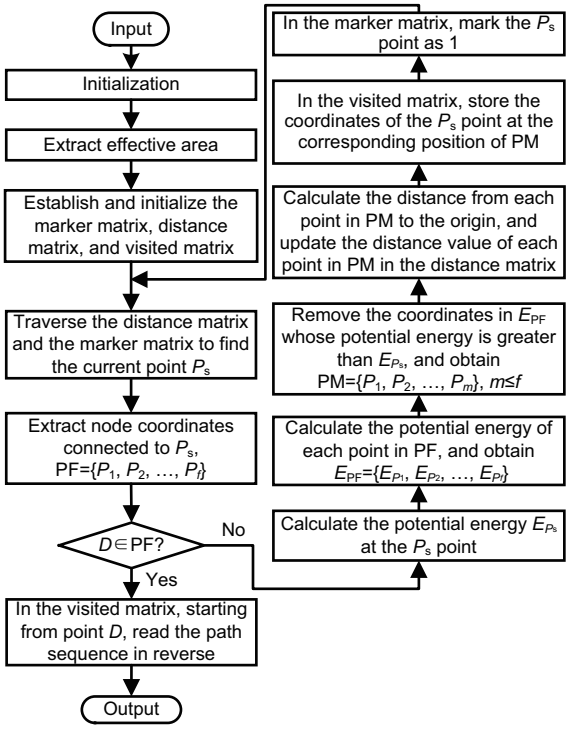

Fig. 9 Overview of APFD

Algorithm 3 Route calculation

Require: map node coordinates
1: Extract the effective area Ω
2: **for** $i \leftarrow 1, 2, \cdots, n_1 - 1$ **do**
   // n_1 is the height
3: **for** $j \leftarrow 1, 2, \cdots, m_1 - 1$ **do**
    // m_1 is the width
4: **if** $i = 0$ && $j = 0$ **then**
5: Mk_max$[i][j] \leftarrow 0$
    // Mk_max is the marker matrix
6: Ds_max$[i][j] \leftarrow 0$
    // Vd_max is the visited matrix
7: Vd_max$[i][j] \leftarrow (x_o, y_o)$
    // Ds_max is the distance matrix
8: **else**
9: Mk_max$[i][j] \leftarrow 0$
10: Ds_max$[i][j] \leftarrow \infty$
11: Vd_max$[i][j] \leftarrow \varnothing$
12: **end if**
13: **end for**
14: **end for**
15: **while** point D does not exist in PF **do**
16: **if** point D exists in PF **then**
17: break
18: **end if**
19: $P_s \leftarrow \min(\text{Ds\_max})$ && $\text{Mk\_max}(x_{P_s}, y_{P_s}) \neq 1$
20: Extract all the nodes connected with P_s and obtain PF $= \{P_1, P_2, \cdots, P_f\}$
21: Calculate E_{P_s} and E_{PF}
22: Remove redundant points to obtain PM $= \{P_1, P_2, \cdots, P_m\}$, $m \le f$
23: Calculate $L_1 = \{l_1^1, l_2^1, \cdots, l_m^1\}$
24: Add value L to each distance value in L_1 to obtain $L_2 = \{l_1^2, l_2^2, \cdots, l_m^2\}$
25: Let the distance value at the PM be $L_3 = \{l_1^3, l_2^3, \cdots, l_m^3\}$
26: **if** $l_i^2 < l_i^3$ **then**
27: Replace l_i^3 with l_i^2 in L_3
28: Vd_max(PM) $\leftarrow P_s$
29: Mk_max$(P_s) \leftarrow 1$
30: **end if**
31: **end while**
32: Obtain the optimal route sequence $Q(O, Q_1, Q_2, \cdots, Q_k, D)$
Ensure: Q

5 Performance evaluation

In this section, we validate the performance of the proposed APFD method and then report the experimental results.

In the simulation and real-word maps, the APFD, AC, A*, RRT, APF, NSGA-II, PSO, and Dijkstra methods are first evaluated for the shortest route, and then the experimental results are analyzed in detail.

5.1 Experimental setup

An Intel® Core™ i7-5557U CPU 3.1 GHz with 4 GB memory is employed on the experimental platform. The number of nodes in the simulation map is $20 \times 40 = 800$. The construction method of the simulation map is shown as follows: starting from the upper left, from left to right, from top to bottom. The horizontal and vertical distance between these nodes is 100 ± 40. The distances between the four vertices of the simulation map and the upper left vertex are $\{0 \pm 40, 0 \pm 40\}$, $\{0 \pm 40, 3900 \pm 40\}$, $\{1900 \pm 40, 0 \pm 40\}$, and $\{1900 \pm 40, 3900 \pm 40\}$, and we mark the coordinates of each network node. Considering the upper left vertex as the origin, the horizontal coordinate increases by 1 from left to right, and the vertical coordinate increases by 1 from top to bottom. The four vertex coordinates are $(0, 0)$, $(0, 39)$, $(19, 0)$, and $(19, 39)$, and the simulation map generated is illustrated in Fig. 10.

Moreover, we employ the GPS trajectory data (about 50 GB) generated by 12 000 taxis in Beijing in November 2012 (http://www.datatang.com) to calculate the actual distance between road network nodes on the Fourth Ring Road in Beijing in the extensive experiments. Each record of the GPS trajectory data includes longitude, latitude, and speed. We use the longitude and latitude values of each data record to locate the taxi position, and obtain the longitude and latitude values when a taxi passes through two road network nodes. The spherical distance formula and the longitude and latitude values of two road network nodes are used to calculate the actual distance between adjacent road network nodes.

5.2 Evaluation metrics

This work defines the performance improvement and efficiency improvement as metrics representing the shortest route and execution efficiency, respectively.

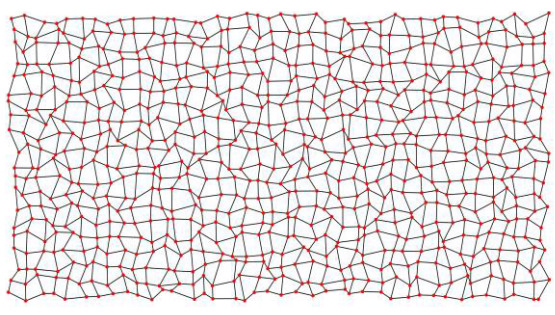

Fig. 10 Simulation map

5.2.1 Performance improvement

Performance improvement is used to evaluate the shortest route of the algorithm. The larger the percentage of performance improvement, the better the performance of the algorithm. It is defined as

$$e = \frac{|l_{\mathrm{m}} - l_{\mathrm{t}}|}{l_{\mathrm{m}}} \times 100\%, \tag{13}$$

where l_{m} is the shortest route value of the APFD method, and l_{t} is the route value of the compared algorithm.

5.2.2 Efficiency improvement

The efficiency improvement is defined to evaluate the efficiency. The larger the multiple of efficiency improvement, the higher the execution efficiency of the algorithm. It is defined as

$$\eta = T_{\mathrm{t}}/T_{\mathrm{m}}, \tag{14}$$

where T_{m} is the shortest time consumed when the APFD method produces the shortest route, and T_{t} is the time consumed by the compared algorithm.

5.3 Experimental results

Twenty pairs of origin and destination points are randomly selected in the simulation map, and the shortest route and efficiency of the APFD method are evaluated in the experiments. The random coordinate points are shown in Table 1.

5.3.1 Route recommendation evaluation

In the simulation map, the APFD, AC, A*, APF, RRT, NSGA-II, and PSO methods are employed for route planning with 20 pairs of random origins and destinations. The results are shown in Table 2 and Fig. 11.

Table 2 shows the total distance value of the shortest route of each method. From Table 2 and Fig. 11, it can be seen that for any pair of origin and destination points, the shortest total distance of the APFD method is smaller than those of the AC, A*, APF, RRT, NSGA-II, and PSO methods. The performance improvement percentages of APFD compared with AC, A*, APF, RRT, NSGA-II, and PSO are reported in Table 3. From

Table 3, compared with AC, A*, APF, RRT, NSGA-II, and PSO, the route planning ability of APFD is improved by 1.45%–39.56%, 4.64%–54.75%, 8.59%–37.25%, 5.06%–45.34%, 0.94%–20.40%, and 2.43%–38.31%, respectively. As depicted in Fig. 12, we select the route diagram of each method at origin (01, 34) and destination (11, 01) for route analysis.

As can be seen from the experimental results, the path planning effects of AC, A*, APF NSGA-II, and PSO algorithms are significantly worse than that of the APFD algorithm. The path planning effect of the RRT algorithm is similar to that of the PAFD algorithm. From Table 2, the total distances of APFD, AC, A*, APF, RRT, NSGA-II, and PSO are 3897, 4244, 5370, 4471, 4094, 4007, and 4314, respectively. The total distance of APFD is obviously smaller than those of AC, A*, APF, RRT, and PSO.

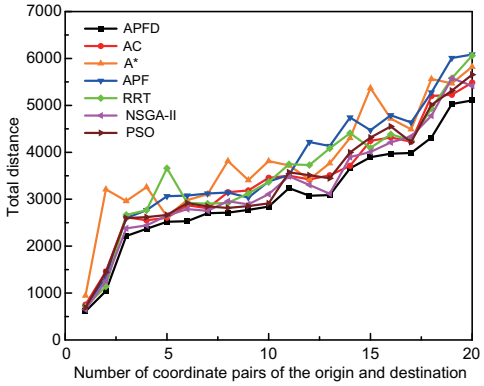

Fig. 11 Comparisons of the shortest route of each method with random coordinates in the simulation map

Table 1 Random coordinate points

Origin and destination points	Coordinate pair	Origin and destination points	Coordinate pair
(00, 05) (10, 28)	A01, B01	(09, 09) (19, 36)	A11, B11
(01, 34) (11, 01)	A02, B02	(01, 04) (19, 15)	A12, B12
(10, 15) (02, 38)	A03, B03	(14, 36) (01, 03)	A13, B13
(00, 00) (19, 39)	A04, B04	(12, 35) (05, 12)	A14, B14
(18, 28) (01, 02)	A05, B05	(17, 03) (03, 38)	A15, B15
(18, 30) (00, 07)	A06, B06	(07, 08) (00, 36)	A16, B16
(09, 03) (15, 27)	A07, B07	(03, 15) (11, 10)	A17, B17
(00, 30) (17, 01)	A08, B08	(00, 05) (10, 28)	A18, B18
(00, 39) (19, 00)	A09, B09	(12, 03) (08, 10)	A19, B19
(16, 03) (08, 21)	A10, B10	(15, 13) (05, 28)	A20, B20

Table 2 Total distance of each method in the simulation map

Origin and destination points	Coordinate pair	Total distance						
		APFD	AC	A*	APF	RRT	NSGA-II	PSO
(12, 03) (08, 10)	A19, B19	610	753	944	703	708	650	689
(03, 15) (11, 10)	A17, B17	1044	1457	1610	1343	1141	1257	1444
(15, 13) (05, 28)	A20, B20	2216	2617	2962	2598	2667	2379	2596
(16, 03) (08, 21)	A10, B10	2368	2551	3258	2767	2765	2442	2618
(00, 05) (10, 28)	A1, B1	2519	2605	2636	3065	3661	2650	2665
(01, 04) (19, 15)	A12, B12	2531	2872	2981	3079	2922	2789	2917
(12, 35) (05, 12)	A14, B14	2701	2807	3108	3121	2906	2753	2847
(09, 03) (15, 27)	A7, B7	2714	3149	3814	3144	2921	2957	2814
(10, 15) (02, 38)	A3, B3	2774	3188	3406	3035	3123	2889	2854
(00, 05) (10, 28)	A18, B18	2843	3457	3813	3379	3367	3108	2912
(07, 08) (00, 36)	A16, B16	3237	3501	3720	3515	3746	3488	3574
(09, 09) (19, 36)	A11, B11	3074	3421	3423	4219	3727	3303	3511
(18, 30) (00, 07)	A6, B6	3088	3509	3768	4134	4085	3117	3447
(18, 28) (01, 02)	A5, B5	3659	3712	4307	4742	4410	3899	3997
(01, 34) (11, 01)	A2, B2	3897	4244	5370	4471	4094	4007	4314
(14, 36) (01, 03)	A13, B13	3972	4317	4718	4789	4384	4212	4551
(00, 30) (17, 01)	A8, B8	3986	4227	4490	4636	4255	4336	4219
(17, 03) (03, 38)	A15, B15	4314	5199	5563	5274	4934	4779	5012
(00, 00) (19, 39)	A4, B4	5030	5227	5468	6010	5579	5578	5315
(00, 39) (19, 00)	A9, B9	5106	5486	5823	6084	6061	5418	5656

The total distance difference between APFD and NSGA-II is the smallest, and we put the route of APFD and NSGA-II into the same graph, as shown in Fig. 12h. Most of the planned path overlaps are different at A, B, C, and D. From these parts, it is obvious that the route of the APFD method is shorter than that of the NSGA-II method. Therefore, the APFD method is better than the NSGA-II method in terms of the total route. It is not difficult to see that the APFD method has better shortest route planning performance than AC, A*, APF, RRT, NSGA-II, and PSO.

5.3.2 Efficiency evaluation

In the simulation map, we evaluate the execution time of APFD and Dijkstra using 20 pairs of random origin and destination coordinates. The route planning results of these two methods are the same. The execution time is sorted by APFD, and the results are shown in Table 4 and Fig. 13.

As depicted in Table 4 and Fig. 13, using 20 pairs of random origin and destination coordinates, the execution time of the APFD method is shorter than that of the Dijkstra method. In detail, according to Eq. (14), we calculate the efficiency improvement of APFD compared with Dijkstra. In Table 4, compared with the Dijkstra method, the

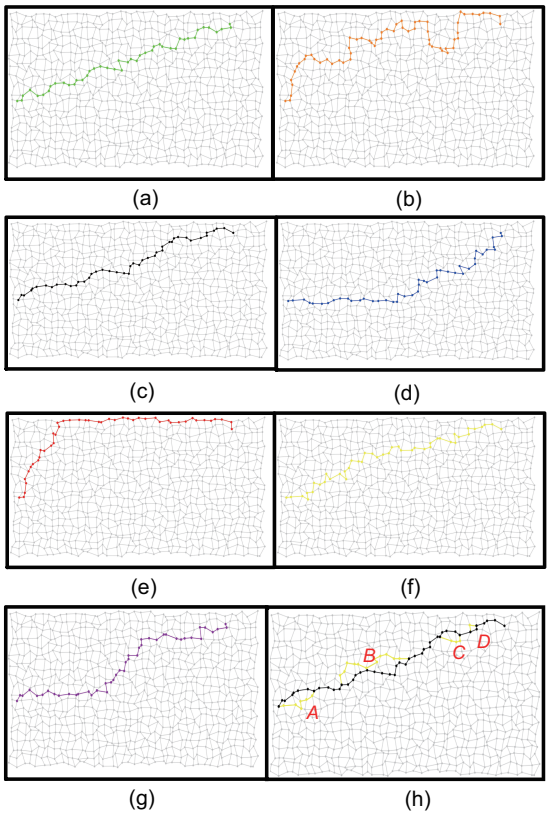

Fig. 12 Comparison of the shortest route of each method with random coordinates: (a) RRT; (b) A*; (c) APFD; (d) APF; (e) AC; (f) NSGA-II; (g) PSO; (h) APFD with NSGA-II

Table 3 Performance improvement of APFD compared with other methods in the simulation map

Origin and destination points	Coordinate pair	Performance improvement of APFD (%)					
		AC	A*	APF	RRT	NSGA-II	PSO
(12, 03) (08, 10)	A19, B19	23.44	54.75	15.25	16.07	6.56	12.95
(03, 15) (11, 10)	A17, B17	39.56	54.21	28.64	9.29	20.40	38.31
(15, 13) (05, 28)	A20, B20	18.10	33.66	17.24	20.35	7.36	17.15
(16, 03) (08, 21)	A10, B10	7.73	37.58	16.85	16.77	3.13	10.56
(00, 05) (10, 28)	A01, B01	3.41	4.64	21.68	45.34	5.20	5.80
(01, 04) (19, 15)	A12, B12	13.47	17.78	21.65	15.45	10.19	15.25
(12, 35) (05, 12)	A14, B14	3.92	15.07	15.55	7.59	1.93	5.41
(09, 03) (15, 27)	A07, B07	16.03	40.53	15.84	7.63	8.95	3.68
(10, 15) (02, 38)	A03, B03	14.92	22.78	9.41	12.58	4.15	2.88
(00, 05) (10, 28)	A18, B18	21.60	34.12	18.85	18.43	9.32	2.43
(07, 08) (00, 36)	A16, B16	8.16	14.92	8.59	15.72	7.75	10.41
(09, 09) (19, 36)	A11, B11	11.29	11.35	37.25	21.24	7.45	14.22
(18, 30) (00, 07)	A06, B06	13.63	22.02	33.87	32.29	0.94	11.63
(18, 28) (01, 02)	A05, B05	1.45	17.71	29.60	20.52	6.56	9.24
(01, 34) (11, 01)	A02, B02	8.90	37.80	14.73	5.06	2.82	10.70
(14, 36) (01, 03)	A13, B13	8.69	18.78	20.57	10.37	6.04	14.58
(00, 30) (17, 01)	A08, B08	6.05	12.64	16.31	6.75	8.78	5.85
(17, 03) (03, 38)	A15, B15	20.51	28.95	22.25	14.37	10.78	16.18
(00, 00) (19, 39)	A04, B04	3.92	8.71	19.48	10.91	10.89	5.67
(00, 39) (19, 00)	A09, B09	7.44	14.04	19.15	18.70	6.11	10.77

execution efficiency of APFD is improved by 1.03–27.75 times. In summary, when the proposed APFD method is compared with the traditional Dijkstra method, the method has greatly improved the execution efficiency.

The reason for the above experimental results is that the APFD method uses the origin and destination coordinates to remove the regions with a high possibility that the optimal path will not appear before path calculation, which greatly reduces the calculation time of the algorithm. In addition, in the path calculation, a potential energy field is established in the road network area, and a critical equipotential line is established according to the current coordinate position and redundant points (which are removed by comparing the potential energy of each connection point with the critical potential), thereby further improving the execution efficiency of APFD. Therefore, compared with Dijkstra, APFD is exponentially improved in terms of execution efficiency.

Table 4 Efficiency comparison between APFD and Dijkstra in the simulation map

Origin and destination points	Coordinate pair	Execution time (ms)		η
		APFD	Dijkstra	
(12, 03) (08, 10)	A19, B19	4	111	27.75
(03, 15) (11, 10)	A17, B17	20	283	14.15
(15, 13) (05, 28)	A20, B20	85	822	9.67
(16, 03) (08, 21)	A10, B10	81	616	7.60
(00, 05) (10, 28)	A1, B1	150	732	4.88
(01, 04) (19, 15)	A12, B12	173	684	3.95
(12, 35) (05, 12)	A14, B14	94	785	8.35
(09, 03) (15, 27)	A7, B7	79	779	9.86
(10, 15) (02, 38)	A3, B3	111	1071	9.65
(00, 05) (10, 28)	A18, B18	153	793	5.18
(07, 08) (00, 36)	A16, B16	129	1020	7.91
(09, 09) (19, 36)	A11, B11	211	1086	5.15
(18, 30) (00, 07)	A6, B6	363	1058	2.91
(18, 28) (01, 02)	A5, B5	426	1079	2.53
(01, 34) (11, 01)	A2, B2	284	1013	3.57
(14, 36) (01, 03)	A13, B13	43	1083	25.19
(00, 30) (17, 01)	A8, B8	535	1085	2.03
(17, 03) (03, 38)	A15, B15	537	1117	2.08
(00, 00) (19, 39)	A4, B4	1053	1116	1.06
(00, 39) (19, 00)	A9, B9	1093	1123	1.03

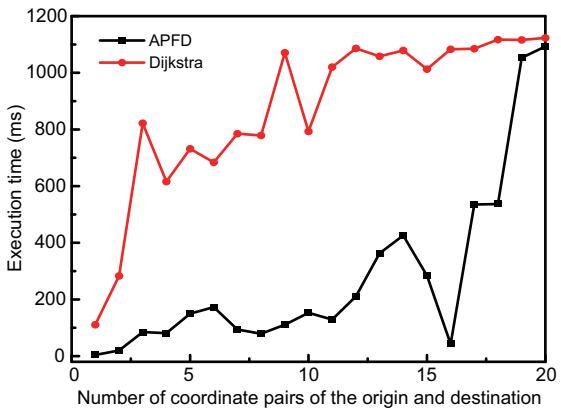

Fig. 13 Execution time comparison between APFD and Dijkstra in the simulation map

5.4 Case study

To verify the effectiveness of the APFD method in real scenarios, we select the road network on the Fourth Ring Road in Beijing for performance evaluation. The method evaluates the shortest route and efficiency to test the effectiveness of the APFD method in the real-world road network.

5.4.1 Route recommendation accuracy

We use 20 pairs of random origin and destination coordinates from the real-world road network in Fig. 14 and conducted route planning experiments using APFD, AC, A*, APF, RRT, NSGA-II, and PSO methods to evaluate the route planning ability. The total distance of route planning is sorted by destination. The comparison of the total experimental distance is shown in Table 5 and Fig. 15, and

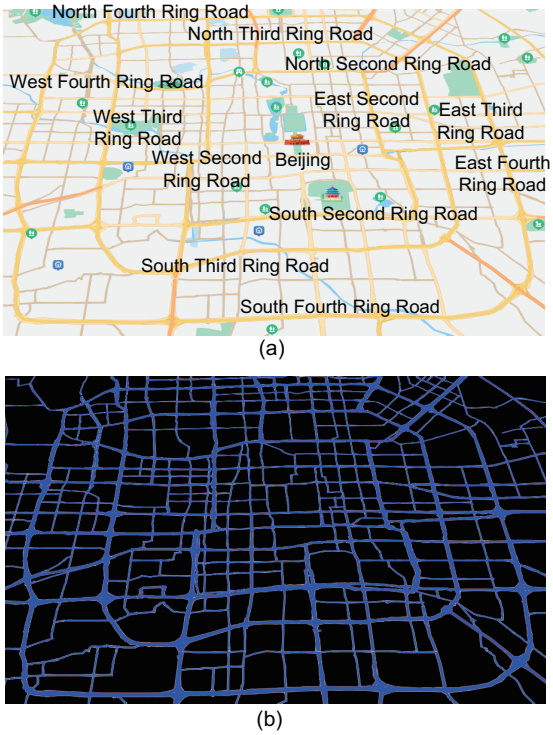

Fig. 14 Real-world road network: (a) road network on the Fourth Ring Road in Beijing; (b) extracted road network

the performance improvement of APFD compared to AC, A*, APF, RRT, NSGA-II, and PSO methods is shown in Table 6. The best route of APFD using one origin and destination pair is illustrated in Fig. 16.

Table 5 shows the total distance of the shortest route in the real application. It can be seen that the shortest total distance of the APFD method is shorter than those of AC, A*, APF, RRT, NSGA-II, and PSO methods in all random coordinate points. Using Eq. (13), we calculate the performance improvement of APFD compared to AC, A*, APF, RRT, NSGA-II, and PSO in Table 6. The results demonstrate that the route planning capability of APFD is improved

Table 5 Total distance of each method in the real-word map

Coordinate pair	Total distance (m)						
	APFD	AC	A*	APF	RRT	NSGA-II	PSO
A15, B15	5697	9415	10 525	8879	7530	6540	5914
A6, B6	5962	6555	7055	7148	8636	6418	7214
A2, B2	8271	9293	10 098	9911	9519	8596	9317
A20, B20	10 604	11 321	12 029	11 998	14 031	12 550	11 178
A12, B12	10 650	11 650	13 151	12 244	13 002	11 473	12 487
A3, B3	12 439	12 947	14 920	13 259	13 512	13 369	13 889
A5, B5	12 444	13 640	15 894	14 889	14 412	12 967	14 187
A17, B17	12 682	13 278	15 451	16 589	17 771	13 131	14 789
A10, B10	13 055	13 920	14 569	15 598	18 879	13 996	14 501
A11, B11	13 514	16 026	18 286	16 519	16 747	14 140	15 547
A4, B4	14 959	16 752	17 002	15 994	16 687	15 887	16 667
A16, B16	15 342	16 231	17 309	16 991	19 002	15 998	17 417
A19, B19	15 542	16 011	16 911	17 017	17 809	16 646	16 468
A7, B7	17 012	17 977	18 388	18 014	19 998	17 579	18 789
A9, B9	19 660	24 265	25 507	24 488	25 916	21 121	23 896
A8, B8	22 401	24 753	33 086	28 146	27 115	23 540	24 789
A1, B1	24 159	24 336	28 391	26 889	26 110	24 878	25 501
A13, B13	26 644	30 036	31 724	28 985	29 936	27 187	29 879
A18, B18	27 321	30 745	35 268	33 132	35 957	28 137	30 024
A14, B14	28 088	32 674	33 733	29 994	32 110	29 898	33 217

Table 6 Performance improvement of APFD compared with other methods in the real-word map

Coordinate pair	Performance improvement of APFD (%)					
	AC	A*	APF	RRT	NSGA-II	PSO
A15, B15	39.49	84.75	55.85	32.17	14.80	3.81
A6, B6	9.05	18.33	19.89	44.85	7.65	21.00
A2, B2	11.00	22.09	19.83	15.09	3.93	12.65
A20, B20	6.33	13.44	13.15	32.32	18.35	5.41
A12, B12	8.58	23.48	14.97	22.08	7.73	17.25
A3, B3	3.92	19.95	6.59	8.63	7.48	11.66
A5, B5	8.77	27.72	19.65	15.81	4.20	14.01
A17, B17	4.49	21.83	30.81	40.13	3.54	16.61
A10, B10	6.21	11.60	19.48	44.61	7.21	11.08
A11, B11	15.67	35.31	22.24	23.92	4.63	15.04
A4, B4	10.70	13.66	6.92	11.55	6.20	11.42
A16, B16	5.48	12.82	10.75	23.86	4.28	13.52
A19, B19	2.93	8.81	9.49	14.59	7.10	5.96
A7, B7	5.37	8.09	5.89	17.55	3.33	10.45
A9, B9	18.98	29.74	24.56	31.82	7.43	21.55
A8, B8	9.50	47.70	25.65	21.04	5.08	10.66
A1, B1	0.73	17.52	11.30	8.08	2.98	5.55
A13, B13	11.29	19.07	8.79	12.36	2.04	12.14
A18, B18	11.14	29.09	21.27	31.61	2.99	9.89
A14, B14	14.04	20.10	6.79	14.32	6.44	18.26

by 0.73%–39.49%, 8.09%–84.75%, 5.89%–55.85%, 8.08%–44.85%, 2.04%–18.35%, and 3.81%–21.55%, respectively. Fig. 16 shows the best route of the APFD method with a random origin and destination. From the above experimental result analysis, we can see that APFD has better route planning capability than AC, A*, APF, RRT, NSGA-II, and PSO in terms of the total route planning distance.

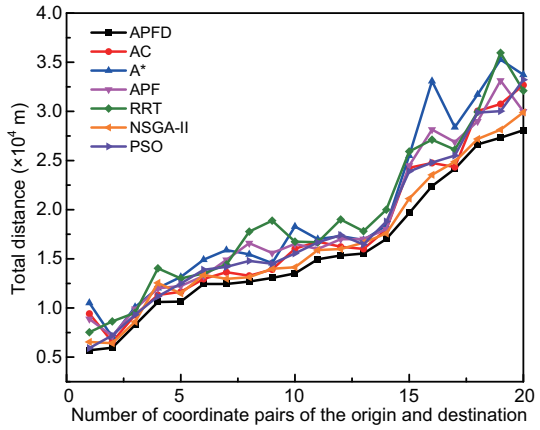

Fig. 15 The shortest route comparison among several methods under random coordinates in the real-word map

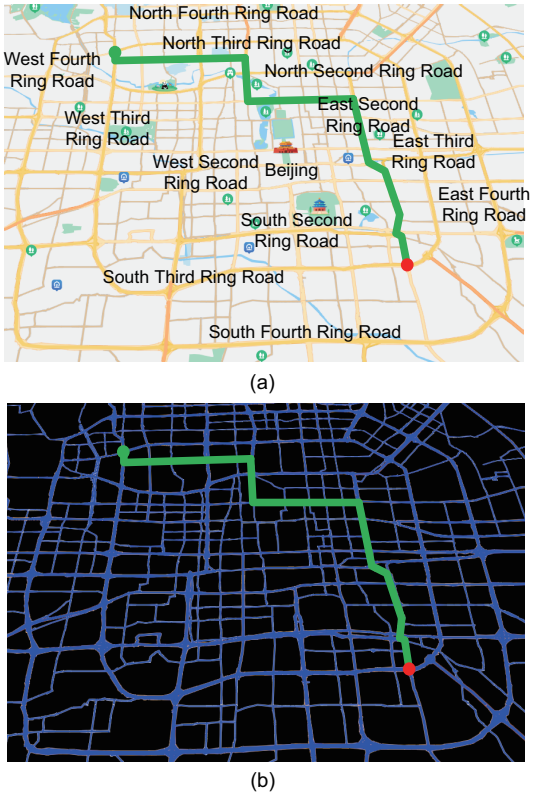

Fig. 16 The shortest route in the real-world road network: (a) road network on the Fourth Ring Road in Beijing; (b) extracted road network

5.4.2 Route recommendation efficiency

In the real map, similar to the simulation map, we compare APFD with Dijkstra on the execution time. The route planning results of these two methods are the same. The execution time sorted by APFD is illustrated in Table 7 and Fig. 17.

As shown in Table 7 and Fig. 17, the execution time of APFD is shorter than that of Dijkstra. Based on Eq. (14), the execution time of APFD is 1.11–8.82 times that of Dijkstra, which is a significant improvement. In summary, execution efficiency has been greatly improved using APFD in the real-road network.

Table 7 Efficiency comparison between APFD and Dijkstra in the real-world map

Number	Coordinate pair	Execution time (ms)		η
		APFD	Dijkstra	
1	A15, B15	11	97	8.82
2	A6, B6	11	79	7.18
3	A2, B2	22	87	8.50
4	A20, B20	32	199	6.22
5	A12, B12	55	299	5.44
6	A3, B3	52	211	4.06
7	A5, B5	35	170	4.86
8	A17, B17	70	228	3.26
9	A10, B10	68	190	2.79
10	A11, B11	51	382	7.49
11	A4, B4	111	358	3.23
12	A16, B16	87	343	3.94
13	A19, B19	84	388	4.62
14	A7, B7	101	402	3.98
15	A9, B9	130	394	3.03
16	A8, B8	242	437	1.81
17	A1, B1	289	459	1.59
18	A13, B13	190	458	2.41
19	A18, B18	406	451	1.11
20	A14, B14	411	485	1.18

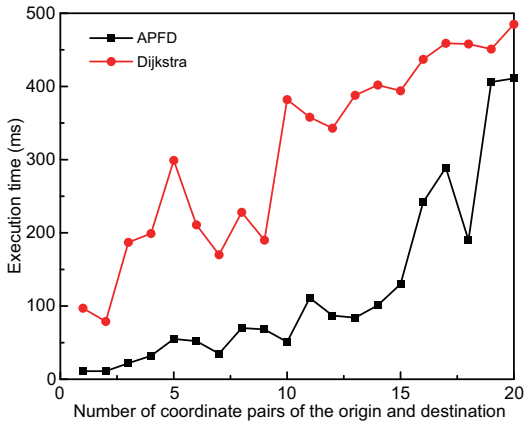

Fig. 17 Comparisons of the shortest route between APFD and Dijkstra in the real-world map

We randomly select 20 pairs of origin and destination coordinates on the simulation map and real-word map. From the experimental results, the route planning ability of the APFD method is significantly better than those of AC, A*, APF, RRT, NSGA-II, and PSO methods. Furthermore, APFD has a shorter execution time compared with the traditional Dijkstra method.

6 Conclusions and future work

To quickly and efficiently recommend optimal route for taxis, we proposed an efficient route recommendation method (named APFD) based on the APF and Dijkstra methods. We first extracted the regions in which the optimal route via the origin and destination would likely be found, and then used the APF method to remove network nodes that might not form the optimal route, namely, redundant points. Finally, the Dijkstra method was used to complete the optimal route recommendation. To validate the effectiveness of the proposed APFD method, we employed a simulation map and a real-word road network map on the Fourth Ring Road in Beijing. In these two maps, APFD, AC, A*, APF, RRT, NSGA-II, PSO, and Dijkstra methods were compared on route planning capabilities. For the comparison of the execution time of the above methods, in the simulation map and real-word map, we randomly chose 20 pairs of origin and destination coordinates as the research objects for route planning. In the simulation map, compared with AC, A*, APF, RRT, NSGA-II, and PSO, the route planning capability of APFD was improved by 1.45%–39.56%, 4.64%–54.75%, 8.59%–37.25%, 5.06%–45.34%, 0.94%–20.40%, and 2.43%–38.31%, respectively. In terms of execution efficiency, the APFD method proved to be 1.03–27.75 times better compared to the Dijkstra method. In the real-word map, compared to AC, A*, APF, RRT, NSGA-II, and PSO, the route planning capability of APFD was improved by 0.73%–39.49%, 8.09%–84.75%, 5.89%–55.85%, 8.08%–44.85%, 2.04%–18.35%, and 3.81%–21.55%, respectively. Compared with the Dijkstra method, the execution efficiency of the APFD method was enhanced by 1.11–8.82 times. In the simulation map and real-world map, the proposed APFD method was significantly better than AC, A*, APF, RRT, NSGA-II, and PSO in route planning. The execution of our APFD method was greatly more efficient than that of the traditional Dijkstra method.

In future work, we will increase the complexity of the road network to validate the proposed method. We will also add traffic flow parameters in the road network, so that the method can update the optimal route in real time based on the current location of the taxi and the locations of passengers.

Contributors

Wenyong ZHANG and Dawen XIA designed the research. Wenyong ZHANG, Dawen XIA, and Huaqing LI proposed the approach and performed the experiments. Guoyan CHANG, Yang HU, Yujia HUO, and Fujian FENG processed the data. Wenyong ZHANG, Dawen XIA, and Huaqing LI drafted the paper. Wenyong ZHANG, Dawen XIA, Yang HU, Yantao LI, and Huaqing LI revised and finalized the paper.

Compliance with ethics guidelines

Wenyong ZHANG, Dawen XIA, Guoyan CHANG, Yang HU, Yujia HUO, Fujian FENG, Yantao LI, and Huaqing LI declare that they have no conflict of interest.

References

Agarwal D, Bharti PS, 2020. Nature inspired evolutionary approaches for robot navigation: survey. *J Inform Optim Sci*, 41(2):421-436. https://doi.org/10.1080/02522667.2020.1723938

Ajeil FH, Ibraheem IK, Sahib MA, et al., 2020. Multi-objective path planning of an autonomous mobile robot using hybrid PSO-MFB optimization algorithm. *Appl Soft Comput*, 89:106076. https://doi.org/10.1016/j.asoc.2020.106076

Akram M, Habib A, Alcantud JCR, 2021. An optimization study based on Dijkstra algorithm for a network with trapezoidal picture fuzzy numbers. *Neur Comput Appl*, 33(4):1329-1342. https://doi.org/10.1007/s00521-020-05034-y

Algfoor ZA, Sunar MS, Abdullah A, 2017. A new weighted pathfinding algorithms to reduce the search time on grid maps. *Expert Syst Appl*, 71:319-331. https://doi.org/10.1016/j.eswa.2016.12.003

Danilovic M, Vasiljevic D, Cvetic B, 2021. A novel pseudo-polynomial approach for shortest path problems. *Networks*, 78(2):107-127. https://doi.org/10.1002/net.22027

He ZC, Chen KY, Chen XY, 2018. A collaborative method for route discovery using taxi drivers' experience and preferences. *IEEE Trans Intell Transp Syst*, 19(8):2505-2514. https://doi.org/10.1109/TITS.2017.2753468

Hoy M, Matveev AS, Savkin AV, 2015. Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey. *Robotica*, 33(3):463-497. https://doi.org/10.1017/S0263574714000289

Huang Y, Ying JJC, Yu PS, et al., 2021. Dynamic graph mining for multi-weight multi-destination route planning with deadlines constraints. *ACM Trans Knowl Discov Data*, 15(1):3. https://doi.org/10.1145/3412363

Ji SG, Wang ZY, Li TR, et al., 2020. Spatio-temporal feature fusion for dynamic taxi route recommendation via deep reinforcement learning. *Knowl-Based Syst*, 205:106302. https://doi.org/10.1016/j.knosys.2020.106302

Jiao ZQ, Ma K, Rong YL, et al., 2018. A path planning method using adaptive polymorphic ant colony algorithm for smart wheelchairs. *J Comput Sci*, 25:50-57. https://doi.org/10.1016/j.jocs.2018.02.004

Kou CX, Hu DD, Yuan JH, et al., 2020. Bisection and exact algorithms based on the Lagrangian dual for a single-constrained shortest path problem. *IEEE/ACM Trans Netw*, 28(1):224-233. https://doi.org/10.1109/TNET.2019.2955451

Lai YX, Lv Z, Li KC, et al., 2019. Urban traffic Coulomb's law: a new approach for taxi route recommendation. *IEEE Trans Intell Transp Syst*, 20(8):3024-3037. https://doi.org/10.1109/TITS.2018.2870990

Lin BL, Zhao YN, Lin RX, et al., 2021. Integrating traffic routing optimization and train formation plan using simulated annealing algorithm. *Appl Math Modell*, 93:811-830. https://doi.org/10.1016/j.apm.2020.12.031

Mac TT, Copot C, Tran DT, et al., 2016. Heuristic approaches in robot path planning: a survey. *Rob Auton Syst*, 86:13-28. https://doi.org/10.1016/j.robot.2016.08.001

Mavrovouniotis M, Li CH, Yang SX, 2017. A survey of swarm intelligence for dynamic optimization: algorithms and applications. *Swarm Evol Comput*, 33:1-17. https://doi.org/10.1016/j.swevo.2016.12.005

Nazarahari M, Khanmirza E, Doostie S, 2019. Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm. *Expert Syst Appl*, 115:106-120. https://doi.org/10.1016/j.eswa.2018.08.008

Nimmagadda MR, Dattawadkar S, Muthukumar S, et al., 2020. Adaptive directional path planner for real-time, energy-efficient, robust navigation of mobile robots. Proc IEEE Int Conf on Robotics and Automation, p.455-461. https://doi.org/10.1109/ICRA40945.2020.9197417

Parimala M, Broumi S, Prakash K, et al., 2021. Bellman–Ford algorithm for solving shortest path problem of a network under picture fuzzy environment. *Compl Intell Syst*, 7(5):2373-2381. https://doi.org/10.1007/s40747-021-00430-w

Przybylski A, Gandibleux X, 2017. Multi-objective branch and bound. *Eur J Oper Res*, 260(3):856-872. https://doi.org/10.1016/j.ejor.2017.01.032

Qin GY, Li TN, Yu B, et al., 2017. Mining factors affecting taxi drivers' incomes using GPS trajectories. *Transp Res Part C Emerg Technol*, 79:103-118. https://doi.org/10.1016/j.trc.2017.03.013

Qu BT, Yang WX, Cui G, et al., 2020. Profitable taxi travel route recommendation based on big taxi trajectory data. *IEEE Trans Intell Transp Syst*, 21(2):653-668. https://doi.org/10.1109/TITS.2019.2897776

Rizk Y, Awad M, Tunstel EW, 2018. Decision making in multiagent systems: a survey. *IEEE Trans Cogn Dev Syst*, 10(3):514-529. https://doi.org/10.1109/TCDS.2018.2840971

Santos FA, Rodrigues DO, Silva TH, et al., 2018. Context-aware vehicle route recommendation platform: exploring open and crowdsourced data. Proc IEEE Int Conf on Communications, p.1-7. https://doi.org/10.1109/ICC.2018.8422972

Schaller Consulting, 2006. The New York City Taxicab Fact Book. Schaller Consulting, Brooklyn, UK.

Sharma K, Doriya R, 2020. Path planning for robots: an elucidating draft. *Int J Intell Robot Appl*, 4(3):294-307. https://doi.org/10.1007/s41315-020-00129-0

Sinyukov DA, Padir T, 2020. CWave: theory and practice of a fast single-source any-angle path planning algorithm. *Robotica*, 38(2):207-234. https://doi.org/10.1017/S0263574719000560

Wang JK, Meng MQH, 2020. Optimal path planning using generalized Voronoi graph and multiple potential functions. *IEEE Trans Ind Electron*, 67(12):10621-10630. https://doi.org/10.1109/TIE.2019.2962425

Wang JK, Chi WZ, Li CM, et al., 2020. Neural RRT*: learning-based optimal path planning. *IEEE Trans Autom Sci Eng*, 17(4):1748-1758. https://doi.org/10.1109/TASE.2020.2976560

Wang YS, Zheng XP, Chen W, et al., 2021. Robust and accurate optimal transportation map by self-adaptive sampling. *Front Inform Technol Electron Eng*, 22(9):1207-1220. https://doi.org/10.1631/FITEE.2000250

Wu N, Wang JY, Zhao WX, et al., 2019. Learning to effectively estimate the travel time for fastest route recommendation. Proc 28th ACM Int Conf on Information and Knowledge Management, p.1923-1932. https://doi.org/10.1145/3357384.3357907

Wu TQ, Yao M, Yang JH, 2016. Dolphin swarm algorithm. *Front Inform Technol Electron Eng*, 17(8):717-729. https://doi.org/10.1631/FITEE.1500287

Yang SY, Ning LJ, Tong LC, et al., 2021. Optimizing electric vehicle routing problems with mixed backhauls and recharging strategies in multi-dimensional representation network. *Expert Syst Appl*, 176:114804. https://doi.org/10.1016/j.eswa.2021.114804

Zajac S, Huber S, 2021. Objectives and methods in multi-objective routing problems: a survey and classification scheme. *Eur J Oper Res*, 290(1):1-25. https://doi.org/10.1016/j.ejor.2020.07.005

Zhang XJ, Jia W, Guan XM, et al., 2019. Optimized deployment of a radar network based on an improved firefly algorithm. *Front Inform Technol Electron Eng*, 20(3):425-437. https://doi.org/10.1631/FITEE.1800749

Zhang Y, Li LL, Lin HC, et al., 2019. Development of path planning approach using improved A-star algorithm in AGV system. *J Int Technol*, 20(3):915-924. https://doi.org/10.3966/160792642019052003023

Zhu DD, Sun JQ, 2021. A new algorithm based on Dijkstra for vehicle path planning considering intersection attribute. *IEEE Access*, 9:19761-19775. https://doi.org/10.1109/ACCESS.2021.3053169