A Novel Heuristic Method for Emergency Path Planning Based on Dynamic Spatial-Temporal Characteristics Map

Bowen Yang¹, Lei Yuan¹, Jin Yan², Zhiming Ding¹,²,³, *, Zhi Cai¹ and Limin Guo¹

¹Beijing University of Technology, Beijing, China, 100124
²Institute of Software, Chinese Academy of Sciences, Beijing, China, 100190
³Beijing Key Laboratory on Integration and Analysis of Large-scale Stream Data, Beijing, China, 100144

*E-mail: zhiming@iscas.ac.cn

Abstract. Emergency path planning technology is one of the hot research points in intelligent transportation systems. There are many methodologies and applications in emergency path planning. However, due to the complexity of the urban network and crowded road conditions, the difficulty of emergency path planning. The objective of emergency path planning is to get the vehicle out of the emergency areas and to its destination in the shortest time. Road congestion caused by emergency situations in cities directly affects the original road network structure. Then the weight of the original road network is no longer suitable as a basis for path recommendation and the value of edges of weight will change over time. To handle the dynamic road network, a novel situational time-stamp heuristic search algorithm (STH) is introduced for the situation space. This algorithm can effectively solve the problem of diversity of situational networks. STH can build a heuristic that adapts to time changes based on the map refresh time, and ensures that the path given in the time window $T$ is optimal. Moreover, STH can give a pruning strategy according to the search time window $T$, which significantly improves the efficiency of the algorithm. Finally, the path planned by STH is better than the baseline algorithm.

1. Introduction

Recently, many researchers are engaged in two kinds of path planning research. One is emergency path recommendation based on artificial intelligence, for example; Bagheri et al. [1] proposed an artificial immune algorithm (AIA) based on an integrated approach. Fu et al. [2] study based on the chance-constrained programming model of the stochastic road network. Kim et al. [3] proposed an evacuation route planning (ERP) model using the spatial structure of the road network to minimize computing time. Zhang et al. [4] presented a collaborative situation model of people and vehicles. This model is a comprehensive linear model of the optimal design for a large number of mixed flows involve pedestrians and vehicles flow in the detour area. However, these methods do not take into account the real-time performance of road congestion, a certain intersection or section will not be always congested. Therefore, the efficient emergency path planning method needs to provide an optimal path to the users within a given time stamp.

The situational road network is a real-time change of road conditions commonly known. The congestion level of urban road network generally changes with the change of time, meanwhile, the congestion level at the same time is also different due to regional differences.
The goal of a detour to a safe area is to select the area closest to the current vehicle location [5]. Molina et al. [6] studied the path evacuation of multiple vehicles. According to the actual situation of logistics supply, this paper considers the problem of evacuation capability when a disaster occurs. So, this paper designs a heuristic search methodology based on the commuting ability of real road network. The main contributions of this paper are as follows:

- A novel situational road network (SRN) methodology based on road traffic capacity and road grade is designed to calculate, the prior path for vehicles effectively.
- A novel situational time-stamp heuristic search (STH) algorithm based on SRN to plan the optimal path for the detour vehicles from the situation road network map in given query time windows $T$.
- A pruning strategy is proposed according to query time window $T$. For different query time windows $T$, the path given by the algorithm query is also different. An appropriate query time window $T$ is given through experimental verification.

In this paper, the emergency scenario is that after heavy congestion caused by congestion in the city, the scheduling problem of a detour road for the vehicle. This point’s remainder is organized as follows: Section 2 will include literature on key factors of emergency evacuation and optimal solution problem. Section 3, a road model based on the situation network is proposed. Section 4 presents an optimal road query finding algorithm based on heuristic factor. Experimental results and conclusion will be showed in Section 5 and Section 6 respectively.

2. Related work
Urban emergency path planning is a very complex and huge system problem. The diversification of urban roads will also bring unpredictable results to path planning [7]. Cai et al. [8] proposed a vector-based dynamic path planning strategy. At present, there are limited researches on route recommendation based on the situational road network. The road network planning based on the situation is dynamic, and the difficulty of dynamic is that the optimal global solution cannot be found [9, 10]. However, the above methods do not consider the real-time changes of the road network because the situation information is continuously changing. Therefore, the Time factor should be added when path planning at some time-stamp.

The path planning for vehicles in an emergency evacuation is main problem [11]. Sometimes the detour effect of vehicles directly affects the path planning of rescue vehicles. The congestion of the road will directly affect the entry time of rescue vehicles and further affect the rescue effect. Therefore, when an emergency disaster occurs, it is necessary to quickly and effectively evacuate vehicles. For this focus, we propose a novel heuristic factor method for vehicle emergency route planning. Given the start point and end point, STH can plan the maximum capacity path from the starting point to the end point in the current time window.

3. Model
A new model will be introduced in detail to satisfy the changing weight under multiple conditions, which is an emergency path planning situational road network model (SRN). The model can perform integrated weighting calculation based on different situation information on existing roads.

3.1. Situational road network model
A city map can be represented as a vector graph $G = \langle V, E \rangle$, where $V$ is a set of intersections, and $E \in \langle v_i, v_j \rangle$ is a set of ordered pairs of vertices which indicate that there is a path between $v_i$ to $v_j$, it is noting that each path is direction and the time at point $v_i$ is ahead of $v_j$ time. To respect one path $\langle v_s, v_d \rangle$, which means vehicles starts form source point $v_s$ and reach their destination at point $v_d$, with a weight $\omega_{ij} := RS, TC, Len >, \omega_{ij} \geq 0$ and this three tuple make up the road weight of SRN model. $RS$ is a road network average speed in time-stamp with $T$ regarding vertex $i$ to vertex $j$. $TC$ represents the road traffic capacity from vertex $i$ to vertex $j$. $\omega_{ij}$ stand for section of road represented by junction $i$ and junction $j$. For the convenience of calculation, SRN methodology will converts the capacity here into capacity factor $\varepsilon$, detail in the experiment section. $Len$ is road distance.
from $v_i$ to $v_j$. According to the distance and the road network’s speed, we can get the time it takes for a car to drive through this section, called $TR$.

3.2. DPath description

The detour path (as DPath in this paper) will be introduced the following contents. The detour path will be expressed as $DPath := (v_1, v_2, v_3, ..., v_n)$ in this paper. Where $DPath$ is the detour path, and $v_n$ is the intersection ID. The road network can understand it as the vertex of the digraph, $v_n \in V$. $DPath$ is an optimal solution from the given vehicle position to the designated safety area.

There are two kinds of paths: the first part is the normal driving road $P$, Equation 1 is used to calculate the common path model. In this part, users usually use the traditional dynamic shortest path planning methods, such as A*-D, BFS-D, D* [12, 13, 14, 15], etc. However, D* applies to the driverless technology, and its mechanism is a step-by-step query, which is insufficient in breadth and does not meet actual navigation requirements.

$$P < s, d > = MinDistance|P_n < s, v_1, v_2, v_3, ..., v_n, d |$$

(1)

Where $P < s, d >$ is a path representation from source $s$ to destination $d$. $P_n$ represents a normal vertex set, where $v_1, v_2, v_3, ..., v_n$ are the path’s vertices. Under normal path circumstances that the result is usually the shortest distance path.

$$DP < e, s > = MinTime|P_d < e, v_{d1}, v_{d2}, v_{d3}, ..., v_{dn}, s |$$

(2)

$DP < e, s >$ stands for a detour path from emergency areas $e$ to safety areas $s$. $P_d$ expresses a set of detour vertices. where $v_{d1}, v_{d2}, v_{d3}, ..., v_{dn}$ are the detour vertices of the path. In the case of the detour path that the minimum time is chosen.

4. Method

This section will introduce the traditional path planning methodology first, followed by a situational time-stamp heuristic algorithm and the setting of query time window $T$. Finally, a pruning strategy based on the detour path planning.

4.1. A situational time-stamp heuristic algorithm

The search graph needs to query the path according to different times. The characteristic of the situational road network is that the weights of the paths that change with time are different. Therefore, our method needs to query the current road network according to the given time window $T$ and return the $T$’s optimal path. In order to meet the requirements that we will design a search function $F(n)$, i.e. Equation 3.

$$F(n) = G(n) + H(n)$$

$$G(n) = \sum_{i=1,j=i+1}^n TR(v_i,v_j), G(n) \leq T$$

(3)

Where $G(n)$ represents the actual time from the source vertex to the $n$th vertex, i.e. Equation 4. $v_i$ and $v_j$ form the currently queried road and $d$ is destination vertex.

**Lemma 1.** Given a directed connected graph G(V, E), if the two points s, d \in V existing a path, there must have a shortest path SP link two points, i.e. SP takes less time than any other paths.

**Proof.** The OPEN table becomes empty after the graph search, while the vertices in the CLOSED table are vertices that have been extended before the end. Since the graph has solutions $(v_1, v_2, ..., v_n)$, there is a path solution. And since vertex pairs are stored successively within CLOSED, there must be a path solution and it is the optimal solution.

$$TR_{(v_i,v_j)} = \frac{e^{Len(v_i,v_j)}}{R_{(v_i,v_j)}}, i < j \in V$$

(5)

3
where $TR_{i \rightarrow j}$ stands for the actual time when the vehicle passes through two intersections $v_i$ and $v_j$, i.e. Equation 5. $\varepsilon$ is the capacity of the road with value range form (0, 1) (detail in Experiment section), $Len_{i \rightarrow j}$ is the distance from $v_i$ to $v_j$ and $RS_{i \rightarrow j}$ represents a value of the average speed of the road from $v_i$ to $v_j$, the value is extracted from the trajectory dataset.

$$H(n) = \lambda = \frac{ED(n,d)}{VS_{avg}}$$

(6)

Where $\lambda = H(n)$ is a heuristic factor, i.e. Equation 6. A heuristic algorithm is proposed that called the situational time-stamp heuristic algorithm (STH) based on SRN model. Equation 6 uses Euclidean divided by the average speed $VS_{avg}$ of the vehicle.

It is worth pointing out that the STH method selects a road based on the sum of the actual time of the road plus the estimated time. The value of $G(n)$ guarantees that the elapsed time of the vehicle is the smallest of all candidate path usage times, which is guaranteed by the principle of BFS search, i.e. $G(n) \leq G(n)^*$. The heuristic function $H(n)$ is monotonically decreasing.

**Lemma 2.** Given a query time window $T$, heuristic path planning methodology is used to find the shortest path subset within time window $T$ in $G$, and when $H(n) \leq H(n)^*$, the more close to the target vertex, the fewer vertices are extended.

**Proof.** According to Lemma 1, there must be a shortest path between any two vertices in a directed connected graph $G$. Therefore, the subset of the most path must be found within a given query time $T$. Using the heuristic principle, the closer the search vertex is to the target vertex, the smaller the value of $H(n)$ will be. And when $H(n) \leq H(n)^*$, it satisfies the monotonic decreasing property, so in the process of searching for vertices, the repeated search of vertices is prevented. Therefore, the smaller $H(n)$ that the fewer number of vertices to explore.

$$ED = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$$

(7)

Where $ED$ is the Euclidean distance between two points $m$ to $d$, $x$ and $y$ represent latitude and longitude respectively, i.e. Equation 7. we will use the estimated time as the heuristic because vehicles always want to reach their destination in the shortest time.

We will perform a BFS search process with $G(n)$ function [16, 17]. If only one vertex is searched down at a time in this way, the method becomes A* search mode and the searching efficiency is not better. In the STH search process, that only limit onetime window $T$. Determine the stopping condition of $G(n)$ by the following Equation 8:

$$G(n) = \begin{cases} 
\sum TR_{v_i,v_j} < T, \\
\sum TR_{v_i,v_j} = T, G(n) = TR_{v_i,v_j} \\
\sum TR_{v_i,v_j} > T 
\end{cases}$$

(8)

It is worth noting that when $TR$’s value in the $G(n)$ search is less than $T$, the search will continue. The search will stop when $TR$ is larger than $T$. In this status, the STH must return the previous vertex because the GPS may not be able to collect the vehicle’s current position accurately when the vehicle is driving between two intersections [18], such as the vehicle entering a tunnel or driving under a bridge. Lines 6-11 of algorithm 1 describes the calculation process of Equation 8.

Figure 1 (a) and (b) show the existing heuristic search algorithm and the BFS algorithm [19, 20], respectively. The solid black line in the figure represents the result of the algorithm query. The dotted line represents the result of the Relaxed edge of the BFS algorithm. A light gray line represents was it without searching edge. Figure 1 (c) shows the STH algorithm with heuristic factors. From Figure 1 show that the BFS search finds more vertices and edges than the heuristic search algorithm. However, STH can relax fewer edges and calculation spending.
Figure 1. Shortest Path Search Process. (a) Shows the A*-D heuristic search algorithm state. (b) Shows the BFS-D search state. (c) Shows the STH algorithm search state.

Algorithm 1 STH

1: Input: situation road network \( G \), Source \( s \), Destination \( d \)
2: Output: best path \( P \)
3: \( F(n) = G(n) + \lambda(n) \), \( n \in E \), CLOSED = \( \emptyset \)
4: while \( s \) search \( G \) do
5: get edge pair \( (v_\alpha, v_\beta) \), \( (v_\alpha, v_\gamma) \)
6: if \( \omega(v_\alpha, v_\beta) > \omega(v_\alpha, v_\gamma) \) then
7: Priority Queue = \( (v_\alpha, v_\gamma) \)
8: return \( v_\gamma \)
9: else if \( \omega(v_\alpha, v_\beta) < \omega(v_\alpha, v_\gamma) \) then
10: Priority Queue = \( \omega(v_\alpha, v_\beta) \)
11: return \( v_\beta \)
12: end if
13: \( G(\gamma) = \omega(v_\alpha, v_\gamma) \)
14: for Greedy BFS search form \( v_\gamma \) do
15: if search \( T_{ij} \ge T \) then
16: Prune \((v_i, v_j)\)
17: break
18: else
19: \( F(v_i, v_j) = G(v_i, v_j) + \lambda(v_i, v_j) \)
20: \( F(v_{i+1}, v_{j+1}) = G(v_{i+1}, v_{j+1}) + \lambda(v_{i+1}, v_{j+1}) \)
21: compare \( F(v_i, v_j) \) with \( F(v_{i+1}, v_{j+1}) \)
22: return smaller value \( F(v_i, v_j) \)
23: end if
24: CLOSED = \( P(v_i, v_j) \)
25: end for
26: end while
Algorithm 1 illustrates the path planning process of STH. When searching from a vertex, the search is started, and unvisited nodes are accessed. In the worst case, every vertex is visited at least once, and every edge is visited at least once. The worst case happens if a node is searched downward in the search process, all its children have been visited, and then it will fall back. Therefore, the time complex degree is \(O(E)\), and the total time complex degree of the algorithm is \(O(N + E)\).

4.2. Pruning strategy

Next a pruning strategy based on STH will be introduced details. The situational road weight is constantly changing. For example, a vehicle in Figure 2 (a) drives from point Source \(s\) to point Destination \(d\). In Figure 2 (a) shows \(P2 = < s, G, L, M, N, O, d >\), and this path results from the traditional path planning algorithm. The purpose is to find the shortest path in the graph and return the result, where \(P2\) is the \(MinDistance\). The red dotted line in the Figure 2 (a) is the search result of STH. At the same time, we can see that the algorithm will expand points A and P. However, these two vertices will be pruned when searching for the next step, not only because they are far away from the target point, but also because they searched the next step for the nodes that have passed through other paths. Thus, the algorithm will no longer continue to expand A and P. Meanwhile, if the candidate path \(P4\) and \(P5\) have no other branch, it will be pruned from candidate \(P\) set. Lines 14-22 of Algorithm 1 describe the pruning strategy.

It can be seen from Figure 2 (b) that there are two congested intersections in the road network at this time. If the vehicle continues to select the current M and R points to explore the target point, the speed of the vehicle will be greatly reduced. STH will cuts off the extended edge of M to the congested area in the next path planning. It will not expand until the next time window \(T\). STH uses this strategy to reduce the extended edges while ensuring the breadth of the extended edges as many extended edges as possible will bring more choices for path planning. Ultimately, \(P11\) results from what we get, and \(P12\), \(P13\), \(P21\), and \(P22\) will be deleted from the candidate path. The length of the detour path is calculated by Equation 9.

\[
P'_{\text{detour}} = \sum_{i=s}^{n} \text{Len}(v_i, v_{i+1})
\]

\[
T_q = \sum_{i=s}^{u} \frac{\text{Len}(v_i, v_{i+1})}{RS < v_i, v_{i+1} >} \leq T
\]

Where \(P'_{\text{detour}}\) represents a detour path sub set, \(s\) is the starting point, \(u\) is the vertex to be queried, and \(RS\) is the road network speed of a section of road. \(\text{Len}(v_i, v_{i+1})\) is the road length, \((v_i, v_{i+1})\) is the adjust point in map. \(T_q\) is the length of each section of road divided by the average speed of its road, i.e. Equation 10.

5. Experiment

In this section, we will introduce the experimental process and steps in detail. The detour path’s significance lies in the moving vehicle’s adjustment scheme when it perceives that the road ahead is
congested so that the vehicle can bypass the congested area and drive to the destination at the minimum cost.

5.1. Experiment settings
We will conduct three comparison experiments: 1) Set different query time window $T$. We use prune strategy to get the appropriate $T$ through the simulation experiment; 2) To verify the influence of $A^*-D$ and BFS-D on vehicle driving on the condition of good road condition and congestion, compared with $A^*-D$ and BFS-D, according to different path lengths; 3) Show the actual planning path effect of $A^*-D$, BFS-D, and STH algorithms.

5.2. Experiment preparation
Our experiment environment is 16G memory, the operating system is 64-bit Windows 10 and the CPU is Intel i7-8700 @3.30GHz. The algorithm compilation language is Java and MATLAB, the map tool uses MapBox$^1$. Dataset is Beijing city road network from OpenStreetMap$^2$ and contains 1691554 Vertices and 1867633 directed Edges. Road traffic data are from the Beijing Municipal Traffic Management Bureau. Traffic data comes from road monitoring devices.

5.3. Verification
This section describes the validation process for this article. First, the OpenStreetMap dataset can be extracted and processed. Roads accessible to vehicles can be classified into five categories: Unclassified, Tertiary, Primary, and Motorway. Secondly, Table 1 shows the capacity division of urban roads refers to the traffic conditions of roads. These are national standards. CRG is city road grade, its value correspond to SL. SL stands for road speed limit and TC stands for road capacity. For the convenience of calculation that we normalize it and called Pass Coefficient $\varepsilon$, and finally the value of $\varepsilon$ import into Equation 5.

| CRG       | Unclassified | Tertiary | Secondary | Primary | Motorway |
|-----------|--------------|----------|-----------|---------|----------|
| SL(km/h)  | 30           | 40-50    | 60-70     | 80      | 90-120   |
| TC        | 600          | 900      | 1060      | 1600    | 1800     |
| $\varepsilon$ | 0.10       | 0.15     | 0.18      | 0.27    | 0.30     |

We will use $T = 5, 15, 30, 45, 60$ as the time window and use 1000 start and end vertices pairs [21], abbreviation (SEVP) (it is worth noting that 1000 start-stop pairs are chosen here because of the load in the local area of the city and road intersections) in Table 2. RV(avg) represents the average numbers of relaxation vertex in STH with 1000 start-stop pairs vertices, ST(avg) is stands for the STH running time of an iteration, IT(avg) represents the average iteration times for each run of STH.

| STH       | SEVP  | RV(avg) | ST(avg) | IT(avg) |
|-----------|-------|---------|---------|---------|
| $T=5$(min)| 1000  | 22342   | 23.2ms  | 26      |
| $T=15$(min) | **1000** | **34721** | **31.7ms** | **21** |
| $T=30$(min)| 1000  | 68924   | 58.1ms  | 18      |
| $T=45$(min)| 1000  | 101936  | 342.4ms | 9       |
| $T=60$(min)| 1000  | 333414  | 832.3ms | 4       |

$^1$https://www.mapbox.com  
$^2$https://www.openstreetmap.org
From Table 2 can shows the STH algorithm runtime result with different query time windows $T$. When $T=15$ the average number of STH relaxed vertices is average 34721, compared with $T=30$ and $T=45$ minutes. In particular, it’s ten times less than $T=60$. From the result of the $ST(\text{avg})$ shows that the value of the query time window $T$ becomes larger, the running time of STH single time also increases. Although the average number of iterations at $T=15$ is greater than $T=30$, $T=45$ and $T=60$, its single calculation speed is small, so it is acceptable. When $T$, although the number of points expanded each time is small, the number of iterations is larger than that in other cases. Moreover, according to the actual road traffic flow variation, more iterations means a waste of computing resources.

**Table 3.** Comparison the results of vehicle travel time with clear and congestion in different path distances

| Case      | Methods | P1(5.7km) | P2(10.8km) | P3(16.1km) | P4(25.6km) |
|-----------|---------|-----------|------------|------------|------------|
| Clear     | A*-D    | 17.4(min) | 39.2(min)  | 51.3(min)  | 63.5(min)  |
|           | BFS-D   | 16.5(min) | 40.9(min)  | 48.5(min)  | 62.8(min)  |
|           | STH      | **15.1(min)** | **41.8(min)** | **49.6(min)** | **61.6(min)** |
| Congestion| A*-D    | 25.1(min) | 39.2(min)  | 91.3(min)  | 133.8(min) |
|           | BFS-D   | 27.2(min) | 43.2(min)  | 88.7(min)  | 132.9(min) |
|           | STH      | **20.1(min)** | **34.5(min)** | **69.2(min)** | **110.3(min)** |

Table 3 shows the vehicle travel time (the trajectory data dataset is extracted by the Minute in this paper) results of dynamic path planning for different algorithms. We selected four paths $<P_1, P_2, P_3, P_4>$ respectively, 5.7km, 10.8km, 16.1km, and 25.6km. We will test single vehicles on roads of different lengths and densities. In the clear state, we can see that the path planned by the STH algorithm takes about the same time as A*-D and BFS-D. But when congestion occurs, the vehicle when A*-D and BFS-D are used for path planning, the driving time of vehicles is greater than that of STH, and with the length of the side, the driving time increases gradually.

![Figure 3](image1.png)

**Figure 3.** The result of path planning with different from paths.
Figure 3 shows the different path planning results from A*-D, BFS-D, and STH algorithms. In the simulation experiment, twenty-five congested areas were added, which are represented by the red no symbol in the Figure 3. The existing algorithm will plan the road network in advance, so when the road network changes, it cannot effectively avoid road congestion. But our STH method can avoid congested areas. As the distance of the path increases, the more congested areas are circumstances navigable. Avoiding congested areas means cars can drive quickly to their destinations. STH choose $T = 15$ as the time window in this experiment.

6. Conclusion
In conclusion, our proposed SRN model can the existing situational road network while the STH methodology based on SRN model that is compared with the existing dynamic path planning algorithm. The experimental results show that the STH can better find the detour path with the shortest execution time when it encounters an emergency, and the algorithm has good performance. Finally, through the analysis of the vehicle's historical trajectory and the change of the road network speed, it was found that when the query time window $T = 15$ minutes, the algorithm is most suitable for the frequency of the road network situation. In future studies will add more situation information to calculate the path, such as POI, weather, population density.

Acknowledgment
This work is supported by National Key R&D Program of China (No.2017YFC0803300), the Beijing Natural Science Foundation (No.4192004), the National Natural Science of Foundation of China (No.61703013, 91646201, 62072016), the Project of Beijing Municipal Education Commission (No.KM201810005023, KM201810005024).

Reference
[1] Bagheri A, Zandieh M, Mahdavi I and Yazdani M 2010 An Artificial Immune Algorithm for The Flexible Job-shop Scheduling Problem, PP. 533-541
[2] Fu H, Zhang Z and Hu G 2010 Developing a Model for The Emergency Rescue Routing Problem Using Stochastic Programming Theory
[3] Kim S, George B and Shekhar S 2007 Evacuation Route Planning: Scalable Heuristics. Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems. Seattle, Washington
[4] Zhang X and Chang G 2014 An Optimization Model for Guiding Pedestrian-Vehicle Mixed Flows During an Emergency Evacuation. J. Intellig. Transport, Systems, vol. 18, pp. 273-285
[5] Khalid M and Yusof U 2015 An Artificial Immune Approach for Optimizing Crowd Emergency Evacuation Route Planning Problem
[6] Molina J, López-Sánchez A, Hernández-Diaz A and Martinez-Salazar I 2018 A Multi-start Algorithm with Intelligent Neighborhood Selection for solving multi-objective humanitarian vehicle routing problems. Journal of Heuristics, vol. 24, pp. 1-23
[7] Ivan M, Ivica A, Niksa J and Tomis H 2018 Urban Road Infrastructure Maintenance Planning with Application of Neural Networks, vol. 2018, pp. 1-10
[8] Zhi C, Xuerui C, Xing S, Qing M, Limin G and Zhiming D 2020 A Novel Vector Based Dynamic Path Planning Method in Urban Road Network. IEEE Access, vol 8, PP. 9046-9060
[9] Jin L and Zhang C 2019 Process Planning Optimization with Energy Consumption Reduction From a Novel Perspective: Mathematical Modeling and a Dynamic Programming-Like Heuristic Algorithm. IEEE Access, vol. 7, pp. 7381-7396
[10] Lawrence T and Hosein P 2019 Stochastic Dynamic Programming Heuristics for Influence Maximization-revenue Optimization. Int. J. Data Sci. Anal., vol. 8, pp. 1-14
[11] Dantzig G and Ramser R 1959 The Truck Dispatching Problem. Management Science, vol. 6, pp. 80-91
[12] Sun X, Yeoh W and Koenig S 2012 Moving Target D * Lite
[13] Setiawan Y, Pratama P, Sang K, Duy V and Sang B 2014 Experimental Comparison of A* and D* Lite Path Planning Algorithms for Differential Drive Automated Guided Vehicle. Lecture Notes in Electrical Engineering, vol. 282, pp. 555-564

[14] Thomas L, Nico P, Christian B and Katharina M 2017 Dynamic Route Planning with Real-time Traffic Predictions. Inf. Syst., vol 64, pp. 258-265

[15] Wakeel A, Aboelmagd N, Hassanein H and Zorba N 2018 iDriveSense: Dynamic Route Planning Involving Roads Quality Information. IEEE Global Communications Conference, GLOBECOM 2018, Abu Dhabi, pp. 1-6

[16] Dijkstra E 1959 A Note on Two Problems in Connexion with Graphs, vol. 1, pp. 269-271 (1959)

[17] Chakraborty S and Satti S 2019 Space-efficient Algorithms for Maximum Cardinality Search, its applications, and variants of BFS, J. Comb. Optim., vol. 37, pp. 465-481

[18] Gao Y, Wang J, Wu W, Arun K and Lim S 2019 Travel Route Planning with Optimal Coverage in Difficult Wireless Sensor Network Environment. Sensors, vol. 19, pp 18-38

[19] Bonet B and Gener H 2000 Planning as heuristic search: New results. Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science, vol. 1809, pp. 160-372

[20] Yiu Y, Du J and Mahapatra R 2019 Evolutionary Heuristic A* Search: Pathfinding Algorithm with Self-Designed and Optimized Heuristic Function. Int. J. Semantic Computing, vol. 13, pp. 2-23

[21] Rui Y, Zhiming D, Yang C and Fujie R 2019 A Path Planning Model based on Spatio-Temporal State Vector from Vehicles Trajectories. IEEE International Conference on Big Data Analytics. Faculty of Information Technology, Beijing University of Technology, Beijing, China, pp. 216-220