HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF INFORMATION
COMMUNICATION TECHNOLOGY

INTRODUCTION TO ARTIFICIAL INTELLIGNECE
IT3160E
INSTRUCTOR: PROF. PHẠM VĂN HẢI
Class ID: 147838

Picture Fuzzy Incident Graph

Group Members:
Đoàn Tiến Dũng - 20220072
Phạm Trường Sang - 20225996
Đinh Xuân Toàn - 20226067
Trần Huy Hoàng - 20225973

Abstract

This report presents the use of Picture Fuzzy Incident Graph (PFIG) to optimize traffic routes in urban environments. This method integrates Picture Fuzzy Sets to model routes and identify potential congestion points. Algorithms such as A*, BFS, and DFS are applied to determine the optimal route based on criteria such as the shortest travel time, fuel efficiency, and avoiding congestion.

1. Introduction

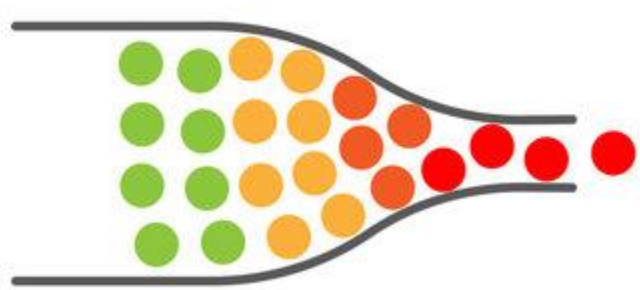
1.1. Problems

Traffic congestion occurs very frequently and poses a significant challenge in urban planning and transportation management. The complexity of urban traffic systems makes it crucial to determine optimal routes for vehicles. This involves considering various criteria such as minimizing travel time, reducing fuel consumption, and effectively avoiding congested areas.

1.2. Main idea

PFIG is a method that uses Picture Fuzzy Sets to evaluate and represent routes and intersections in the graph. This approach helps to handle the uncertainty and multi-valued nature of the data.

After using Picture Fuzzy Incident Graph method, we find PFICP (Picture Fuzzy Incident Connection Path). It is an important route and has a large number of people intending to join it. For easy reference, it is like a bottleneck and the more vehicles join in it, the more congested people get. Finding PFICP plays an important role in reduce traffic jam and boost traffic flow faster.



2. System Data

2.1. Input data

PFS is described by three components: Positive Membership (PM), Neutral Membership (NM), and Negative Membership (nM) of an element, satisfying 0 ≤ PM + NM + nM ≤ 1. The sum of it greater than or equal to 0 and less than or equal to 1.

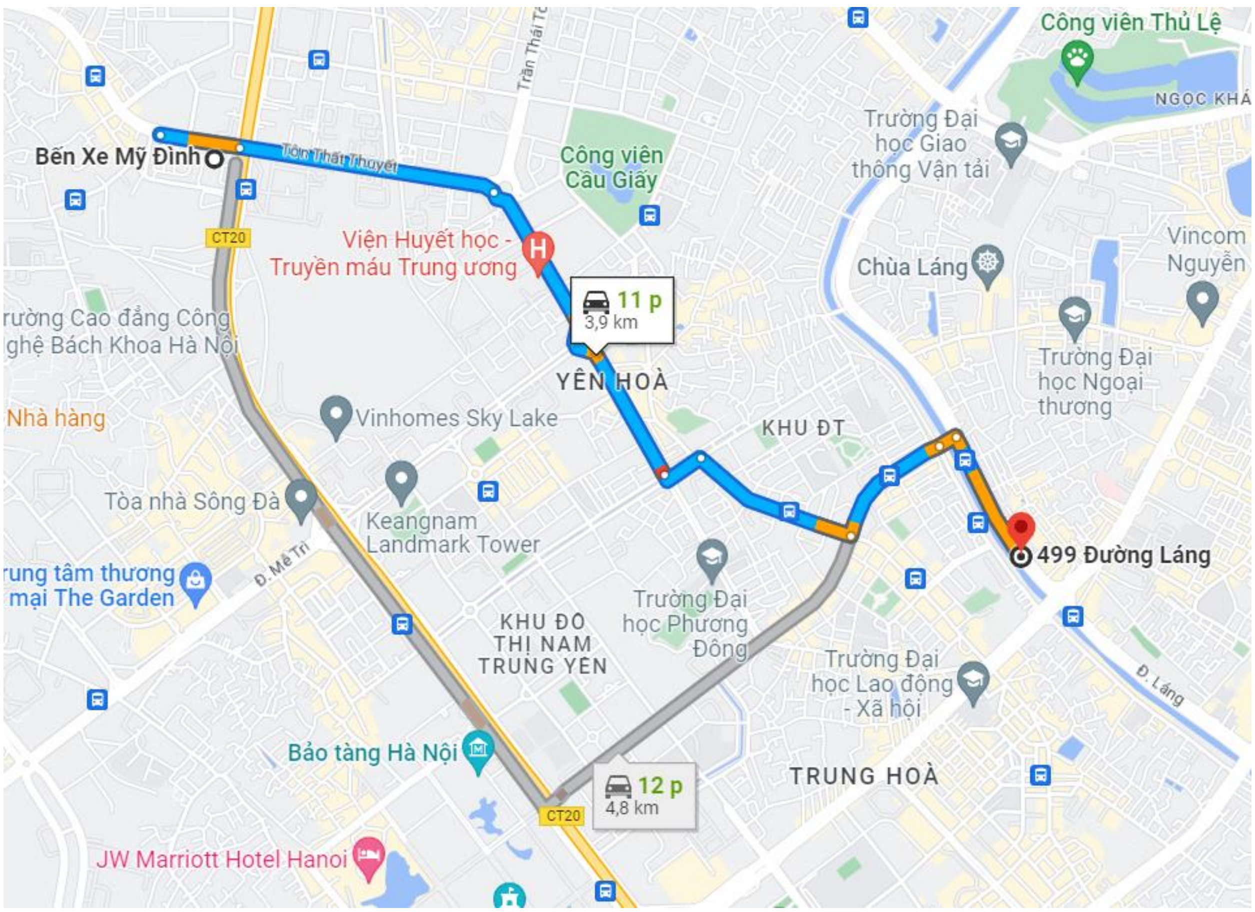
For example, with this data to the destination Hust:

Mỹ Đình, Mỹ Đình – Nguyễn Chí Thanh

P_M = 0.1

N_M = 0.2

n_M = 0.3



The route from My Dinh bus station to Nguyen Chi Thanh Road

We can understand that is:

- P_M= 0.1, representing the percentage of people traveling from My Dinh to Nguyen Chi Thanh or vice versa who intend to reach HUST.

- N_M = 0.2, indicating the percentage of people who are unsure whether to take the My Dinh - Nguyen Chi Thanh route to reach HUST.
- n_M= 0.3, indicating the percentage of people not planning to take the My Dinh.

- Nguyen Chi Thanh route to reach HUST.

2.2. Output data:

Which routes are PFICPs and the optimal route based on the evaluations of the algorithms and the PFS values.

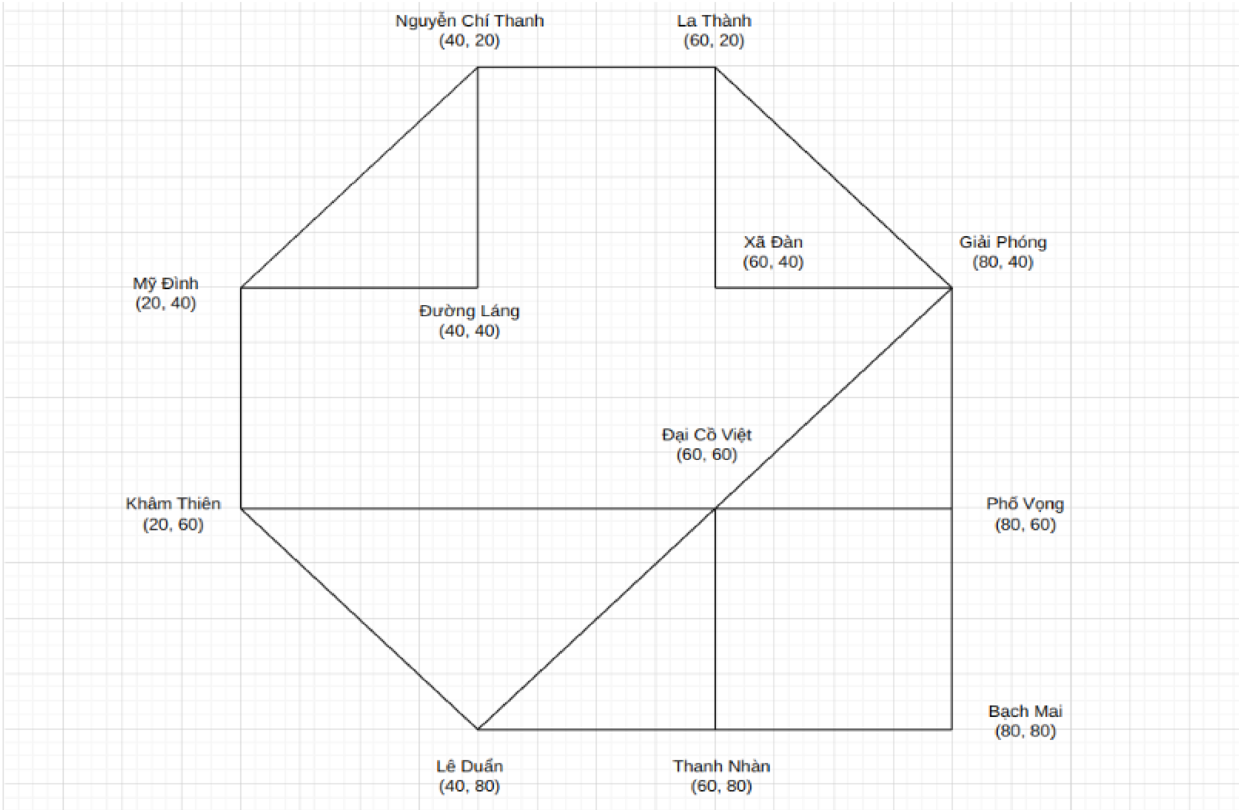
3. AI model / Algorithms

Step 1: Define the Problem:

Clearly define the criteria to determine the "best" route from point A to B: minimum travel time, low fuel consumption, avoiding traffic congestion.

Step 2: Presentation:

Describe the problem in the form of a graph, with vertices representing streets and edges representing routes.



Step 3: Build Fuzzy Incidence Matrices:

For each vertex and edge in the graph, construct a fuzzy incidence matrix to describe the relationship between vertices and edges.

```
Key: ('Kham Thien', ('Kham Thien', 'My Dinh')) , value: {'PM': 0.41, 'NM': 0.47, 'nM': 0.06} , Total: 0.94
Key: ('My Dinh', ('Kham Thien', 'My Dinh')) , value: {'PM': 0.12, 'NM': 0.59, 'nM': 0.17} , Total: 0.88
Key: ('Dai Co Viet', ('Dai Co Viet', 'Giai Phong')) , value: {'PM': 0.39, 'NM': 0.51, 'nM': 0.08} , Total: 0.98
Key: ('Giai Phong', ('Dai Co Viet', 'Giai Phong')) , value: {'PM': 0.02, 'NM': 0.33, 'nM': 0.59} , Total: 0.94
Key: ('Giai Phong', ('Giai Phong', 'Pho Vong')) , value: {'PM': 0.12, 'NM': 0.14, 'nM': 0.74} , Total: 1.0
Key: ('Pho Vong', ('Giai Phong', 'Pho Vong')) , value: {'PM': 0.06, 'NM': 0.33, 'nM': 0.42} , Total: 0.81
Key: ('Dai Co Viet', ('Dai Co Viet', 'Kham Thien')) , value: {'PM': 0.44, 'NM': 0.08, 'nM': 0.43} , Total: 0.95
Key: ('Kham Thien', ('Dai Co Viet', 'Kham Thien')) , value: {'PM': 0.51, 'NM': 0.27, 'nM': 0.14} , Total: 0.92
Key: ('Dai Co Viet', ('Dai Co Viet', 'Pho Vong')) , value: {'PM': 0.43, 'NM': 0.44, 'nM': 0.03} , Total: 0.9
Key: ('Pho Vong', ('Dai Co Viet', 'Pho Vong')) , value: {'PM': 0.17, 'NM': 0.27, 'nM': 0.23} , Total: 0.67
Key: ('Kham Thien', ('Kham Thien', 'Le Duan')) , value: {'PM': 0.37, 'NM': 0.27, 'nM': 0.25} , Total: 0.89
Key: ('Le Duan', ('Kham Thien', 'Le Duan')) , value: {'PM': 0.24, 'NM': 0.38, 'nM': 0.3} , Total: 0.92
Key: ('Dai Co Viet', ('Dai Co Viet', 'Le Duan')) , value: {'PM': 0.5, 'NM': 0.24, 'nM': 0.13} , Total: 0.87
Key: ('Le Duan', ('Dai Co Viet', 'Le Duan')) , value: {'PM': 0.0, 'NM': 0.21, 'nM': 0.77} , Total: 0.98
Key: ('Dai Co Viet', ('Dai Co Viet', 'Thanh Nhan')) , value: {'PM': 0.09, 'NM': 0.51, 'nM': 0.1} , Total: 0.7
Key: ('Thanh Nhan', ('Dai Co Viet', 'Thanh Nhan')) , value: {'PM': 0.12, 'NM': 0.37, 'nM': 0.39} , Total: 0.88
Key: ('Bach Mai', ('Bach Mai', 'Pho Vong')) , value: {'PM': 0.2, 'NM': 0.15, 'nM': 0.43} , Total: 0.78
Key: ('Pho Vong', ('Bach Mai', 'Pho Vong')) , value: {'PM': 0.35, 'NM': 0.11, 'nM': 0.19} , Total: 0.65
Key: ('Bach Mai', ('Bach Mai', 'Thanh Nhan')) , value: {'PM': 0.2, 'NM': 0.26, 'nM': 0.28} , Total: 0.74
Key: ('Thanh Nhan', ('Bach Mai', 'Thanh Nhan')) , value: {'PM': 0.22, 'NM': 0.62, 'nM': 0.15} , Total: 0.99
Key: ('Le Duan', ('Le Duan', 'Thanh Nhan')) , value: {'PM': 0.29, 'NM': 0.35, 'nM': 0.36} , Total: 1.0
Key: ('Thanh Nhan', ('Le Duan', 'Thanh Nhan')) , value: {'PM': 0.08, 'NM': 0.48, 'nM': 0.28} , Total: 0.84
```

Step 4: Evaluate and select the Best Route:

At first, use FPIG to determine PFICP and after that add some algorithms to find the best route: A*, BFS, DFS.

A* (A-star)

The A* algorithm is a pathfinding algorithm used to find the shortest path from a start node to an end node in a graph. It uses a cost function that combines the actual cost and the estimated remaining cost to find the optimal path.

BFS (Breadth-First Search) and DFS (Depth-First Search)

- **BFS**: Breadth-First Search, explores all nodes at the present depth level before moving on to nodes at the next depth level.
- **DFS**: Depth-First Search, explores as far down a branch as possible before backtracking.

Step 5: Optimization:

In fact, the data will always change to change in real time, so after we move from one point to another point, the data will be changed.

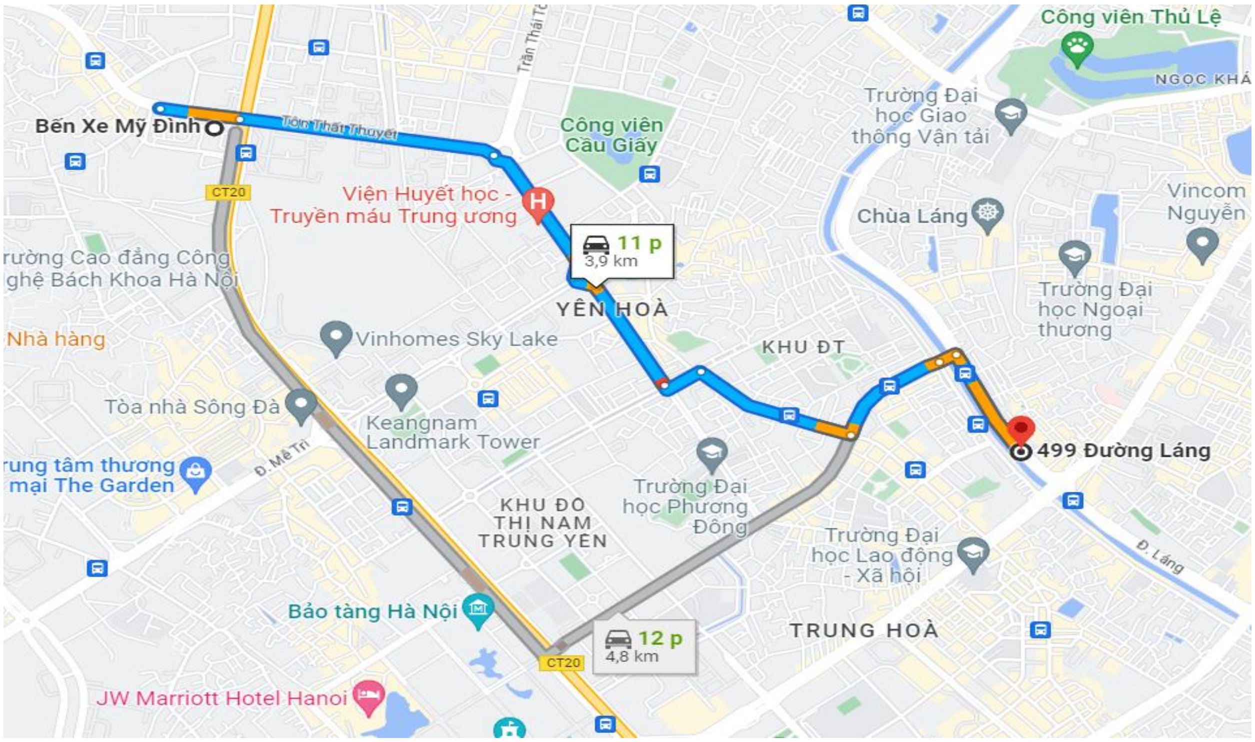
➔

```
Key: ('Kham Thien', ('Kham Thien', 'My Dinh')) , value: {'PM': 0.41, 'NM': 0.47, 'nM': 0.06} , Total: 0.94
Key: ('My Dinh', ('Kham Thien', 'My Dinh')) , value: {'PM': 0.12, 'NM': 0.59, 'nM': 0.17} , Total: 0.88
Key: ('Dai Co Viet', ('Dai Co Viet', 'Giai Phong')) , value: {'PM': 0.39, 'NM': 0.51, 'nM': 0.08} , Total: 0.98
Key: ('Giai Phong', ('Dai Co Viet', 'Giai Phong')) , value: {'PM': 0.02, 'NM': 0.33, 'nM': 0.59} , Total: 0.94
Key: ('Giai Phong', ('Giai Phong', 'Pho Vong')) , value: {'PM': 0.12, 'NM': 0.14, 'nM': 0.74} , Total: 1.0
Key: ('Pho Vong', ('Giai Phong', 'Pho Vong')) , value: {'PM': 0.06, 'NM': 0.33, 'nM': 0.42} , Total: 0.81
Key: ('Dai Co Viet', ('Dai Co Viet', 'Kham Thien')) , value: {'PM': 0.44, 'NM': 0.08, 'nM': 0.43} , Total: 0.95
Key: ('Kham Thien', ('Dai Co Viet', 'Kham Thien')) , value: {'PM': 0.51, 'NM': 0.27, 'nM': 0.14} , Total: 0.92
Key: ('Dai Co Viet', ('Dai Co Viet', 'Pho Vong')) , value: {'PM': 0.43, 'NM': 0.44, 'nM': 0.03} , Total: 0.9
Key: ('Pho Vong', ('Dai Co Viet', 'Pho Vong')) , value: {'PM': 0.17, 'NM': 0.27, 'nM': 0.23} , Total: 0.67
Key: ('Kham Thien', ('Kham Thien', 'Le Duan')) , value: {'PM': 0.37, 'NM': 0.27, 'nM': 0.25} , Total: 0.89
Key: ('Le Duan', ('Kham Thien', 'Le Duan')) , value: {'PM': 0.24, 'NM': 0.38, 'nM': 0.3} , Total: 0.92
Key: ('Dai Co Viet', ('Dai Co Viet', 'Le Duan')) , value: {'PM': 0.5, 'NM': 0.24, 'nM': 0.13} , Total: 0.87
Key: ('Le Duan', ('Dai Co Viet', 'Le Duan')) , value: {'PM': 0.0, 'NM': 0.21, 'nM': 0.77} , Total: 0.98
Key: ('Dai Co Viet', ('Dai Co Viet', 'Thanh Nhan')) , value: {'PM': 0.09, 'NM': 0.51, 'nM': 0.1} , Total: 0.7
Key: ('Thanh Nhan', ('Dai Co Viet', 'Thanh Nhan')) , value: {'PM': 0.12, 'NM': 0.37, 'nM': 0.39} , Total: 0.88
Key: ('Bach Mai', ('Bach Mai', 'Pho Vong')) , value: {'PM': 0.2, 'NM': 0.15, 'nM': 0.43} , Total: 0.78
Key: ('Pho Vong', ('Bach Mai', 'Pho Vong')) , value: {'PM': 0.35, 'NM': 0.11, 'nM': 0.19} , Total: 0.65
Key: ('Bach Mai', ('Bach Mai', 'Thanh Nhan')) , value: {'PM': 0.2, 'NM': 0.26, 'nM': 0.28} , Total: 0.74
Key: ('Thanh Nhan', ('Bach Mai', 'Thanh Nhan')) , value: {'PM': 0.22, 'NM': 0.62, 'nM': 0.15} , Total: 0.99
Key: ('Le Duan', ('Le Duan', 'Thanh Nhan')) , value: {'PM': 0.29, 'NM': 0.35, 'nM': 0.36} , Total: 1.0
Key: ('Thanh Nhan', ('Le Duan', 'Thanh Nhan')) , value: {'PM': 0.08, 'NM': 0.48, 'nM': 0.28} , Total: 0.84
```

Thus, we need to repeat the algorithm at each point in the path we find until reaching the destination.

4. Expected results

Convert the selected route into a format that vehicle owners can understand, such as GPS coordinates or directions on Google Maps.



5. Experiment and evaluation

After experiment and run code, we have a table of time algorithms (with small data sheet):

```
PFICPs: [('Xa Dan', ('Giai Phong', 'Xa Dan')), ('Dai Co Viet', ('Dai Co Viet', 'Giai Phong')), ('Bach Mai', ('Bach Mai', 'Thanh Nhan'))]
With dfs_search, currently at node: My Dinh, moving to node: Duong Lang

Speed: 40 km/h
A* route: ['Bach Mai', 'Pho Vong', 'Giai Phong', 'La Thanh', 'Nguyen Chi Thanh', 'Duong Lang']
Total run time: 2.71 hours. Total route length: 108.28 km. Total search route time: 0.0033860206604003906 s
BFS route: ['Bach Mai', 'Pho Vong', 'Giai Phong', 'La Thanh', 'Nguyen Chi Thanh', 'Duong Lang']
Total run time: 2.71 hours. Total route length: 108.28 km. Total search route time: 0.0006847381591796875 s
DFS route: ['Bach Mai', 'Pho Vong', 'Giai Phong', 'Dai Co Viet', 'Thanh Nhan', 'Le Duan', 'Kham Thien', 'My Dinh', 'Duong Lang']
Total run time: 5.12 hours. Total route length: 176.57 km. Total search route time: 0.0011758804321289062 s
 lits@part-time-introvert  ~/AI prj  master                                                     ✓  35
```

```
Speed: 40 km/h
A* route: ['Bach Mai', 'Pho Vong', 'Dai Co Viet', 'Kham Thien', 'My Dinh', 'Duong Lang']
Total run time: 3.5 hours. Total route length: 120.0 km. Total search route time: 0.003480672836303711 s
BFS route: ['Bach Mai', 'Pho Vong', 'Giai Phong', 'La Thanh', 'Nguyen Chi Thanh', 'Duong Lang']
Total run time: 3.21 hours. Total route length: 108.28 km. Total search route time: 0.0007550716400146484 s
DFS route: ['Bach Mai', 'Thanh Nhan', 'Le Duan', 'Dai Co Viet', 'Pho Vong', 'Giai Phong', 'Xa Dan', 'La Thanh', 'Nguyen Chi Thanh', 'Duong Lang']
Total run time: 6.21 hours. Total route length: 188.28 km. Total search route time: 0.001470327377319336 s
 lits@part-time-introvert  ~/AI prj  master                                               ✓  3510  12:
```

Here is the result when run code 10 times:

Times	Algorithm/Value	Estimate Time(hours)	Total Route Length (km)	Total Search Route Time (s)	Times	Algorithm/Value	Estimate Time(hours)	Total Route Length (km)	Total Search Route Time (s)
1	A*	2.5	12.5	0.45	6	A*	2.5	12.6	0.43
	BFS	2.7	13.0	0.35		BFS	2.7	12.9	0.37
	DFS	3.0	15.0	0.55		DFS	3.1	14.9	0.50
2	A*	2.4	12.3	0.42	7	A*	2.4	12.4	0.46
	BFS	2.8	13.2	0.38		BFS	2.8	13.3	0.35
	DFS	3.1	14.8	0.51		DFS	3.2	15.3	0.52
3	A*	2.6	12.8	0.47	8	A*	2.6	12.9	0.48
	BFS	2.6	12.5	0.34		BFS	2.6	12.6	0.32
	DFS	3.2	15.2	0.58		DFS	3.3	15.0	0.56
4	A*	2.3	12.0	0.40	9	A*	2.3	12.1	0.41
	BFS	2.9	13.5	0.36		BFS	2.9	13.2	0.39
	DFS	3.3	15.5	0.53		DFS	3.1	15.2	0.54
5	A*	2.7	12.7	0.49	10	A*	2.7	12.8	0.50
	BFS	2.5	12.8	0.33		BFS	2.5	12.7	0.34
	DFS	3.0	15.1	0.57		DFS	3.2	15.4	0.58

- Total route length: DFS>A*>=BFS
- Total code run time: A*>DFS>BFS

6. Conclusion

This project aims to find the optimal route for vehicles by using a combination of PFIG, A*, BFS, and DFS algorithms. The expected result is to identify routes with the best criteria such as shortest travel time, least fuel consumption, and avoiding congestion. The experiments and evaluations of algorithm running times will provide detailed insights into the effectiveness of the methods used.

7. Reference

[1] A.Mohamed Ismayil and N.Asha Bosley Domination in Picture Fuzzy Graphs.

[2] B. Cuong and P. Hai, Some fuzzy logic operators for picture fuzzy sets, Seventh International Conference on Knowledge and Systems Engineering (2015) 132-137 .

[3] B. C. Cuong, Picture fuzzy sets, Journal of Computer Science and Cybernetics, 30, No. 4 (2014) 409-420.

[4] Irfan Nazeer and Tabasam Rashid , Picture Fuzzy Incidence Graphs with Application.

[5] Bui-Anh-Tuan-20215251, Picture Fuzzy Incident Graph and its Application in the Problem of Traffic Flow Control on the Hanoi Route.