# **AI Capstone Project – Group 5**

|  |  |  |
| --- | --- | --- |
| Name | Student ID | Mail |
| Tran Vuong Quoc Dat | 20200145 | dat.tvp200145@sis.hust.edu.vn |
| Nguyen Truong Truong An | 20204866 | an.ntt204866@sis.hust.edu.vn |
| Le Duc Anh Tuan | 20204929 | tuan.lda204929@sis.hust.edu.vn |
| Nguyen Nho Trung | 20204894 | trung.nn204894@sis.hust.edu.vn |
| Nguyen Thanh Dat | 20204903 | dat.nt204903@sis.hust.edu.vn |

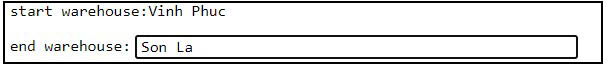
**1. Introduction**

Write a program to find the optimal path between 2 warehouses so that both time and cost are optimized( e.g., warehouse in Hanoi and Vinh city). The warehouse system is divided into two types: main warehouse and secondary warehouse

Goods between any 2 main warehouses can be transported directly by plane, the default speed is v\_plane km/h. Goods between 2 secondary warehouses or between 1 secondary warehouse to 1 main warehouse and vice versa will go by road, the default speed is v\_truck km/h. Every time it passes through a main warehouse, the goods must be stored there for k hours.

**Input:**

* A map of cities in Vietnam with cities and distance between them, and by default, each city will have one warehouse. There are 3 main warehouses, they will be randomly generated and distributed co that Each domain will have 1 main warehouse( E.g: [Ha Noi in the north, Da Nang in the central and Ho Chi Minh in the south]
* file ‘data\_excel.xlsx’:Contains data on the distance between the warehouses according to the crow's flight path
* start warehouse and end warehouse. start and ending warehouse can be either main warehouse or secondary warehouse



Output:

* The path from start warehouse to end warehouse
* The cost parameter represents the optimization function of the path, the smaller this parameter is, the more optimal the path is both in terms of time and transportation cost.



**2. Implementation plan**

**2.1. Pre-processing**

* Collect coordinate data and create a dataset of 64 provinces in Vietnam, then by default, there will be 1 secondary warehouse in each province
* Processing from coordinate data to distance data
* Collect v\_truck: truck speed, v\_plane: plane speed, cost\_truck: truck cost, cost\_plane: plane cost
* Convert distance data into evaluation parameter for each algorithm

A,B,C,D: node represents the warehouses

* AStar:

A dictionary contains nodes, g(x) and h(x)

The data structure:



evaluation function:

f(x) = g(x) + h(x)

Nodes with small f(x) will be given priority

* UCS:

a dictionary contains nodes and g(x)

The data structure:



evaluation function:

f(x) = g(x)

* DFS

a dictionary contains nodes and g(x)

The data structure:



This algorithm doesn’t have an evaluation function

Approved according to FILO rules

* IDA:

Data get directly from excel file

evaluation function:

f(x) = h(x) + g(x)

Parameter formula:

* g(x): cost parameter from node x to node parent

Fomulation: g(x) = (distance/v + storage\_time) \* cost

E.g: g(B) = cost parameter from B to A( A is B’s parent node)

* h(x): cost parameter from node x to node goal

Fomulation: h(x) = (distance/v) \* 2 \* cost

E.g: h(B) = cost parameter from B to E( E is the end warehouse)

By default, between 2 main warehouse: v = v\_plane = 800km/h, cost = cost\_plane = 160.000 vnđ/kg, storage\_time = 2h

In other cases: v = v\_truck = 60km/h, cost = cost\_truck = 35.000 vnđ/kg, storage\_time = 0h

**2.2.Deployment**

**2.2.1. Data constructor programming:**

Create separate data for each algorithm and package it into modules, others just need to import and use

**2.2.2. Programming algorithms**

We program 4 algorithms:

* A\*
* Uniform cost search
* IDA\*
* Depth First Search

**2.2.3. Apply algorithm, use constraints to find the optimal path:**

* If the start warehouse and end warehouse aren’t in the same domain:

Good must be transported to the main warehouse that placed in the same area to start warehouse before transported to end warehouse.

* If the start warehouse and end warehouse aren’t in the same domain:

If the path from the start warehouse to the main warehouse is the longest path of the triangle start warehouse - main warehouse - end warehouse that goods must be transported to the main warehouse before being transported to the end warehouse.

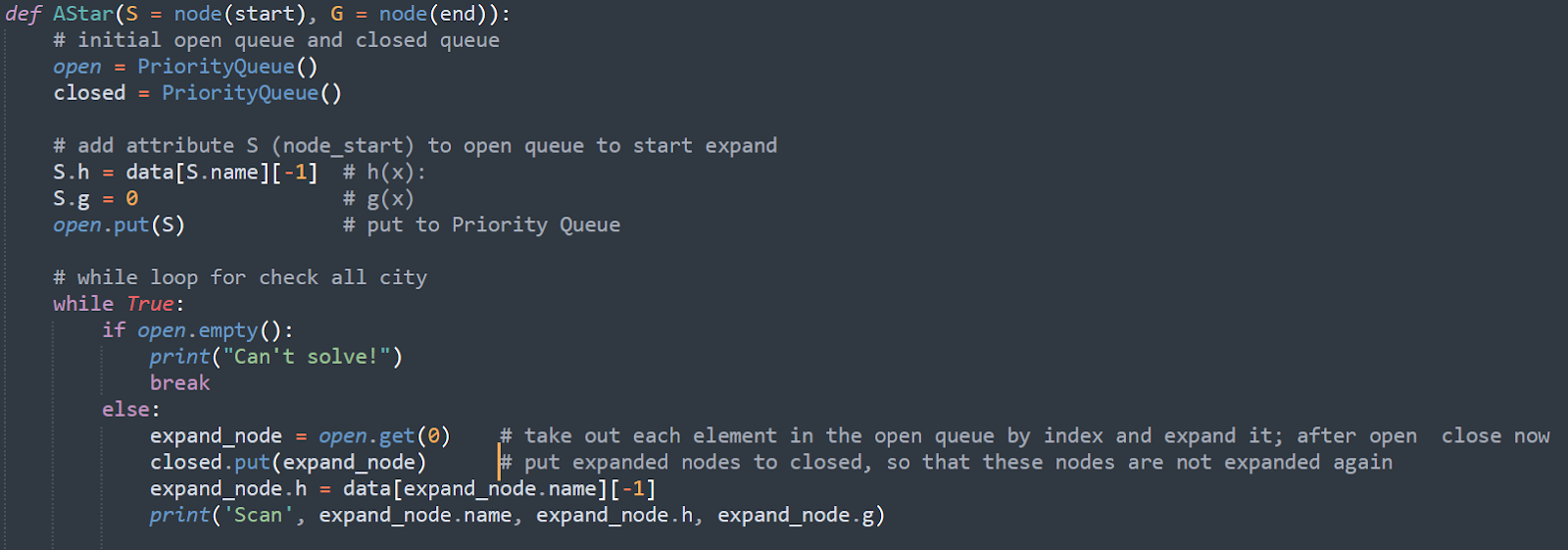
In other cases, goods can be transported directly from the start warehouse to the end warehouse.

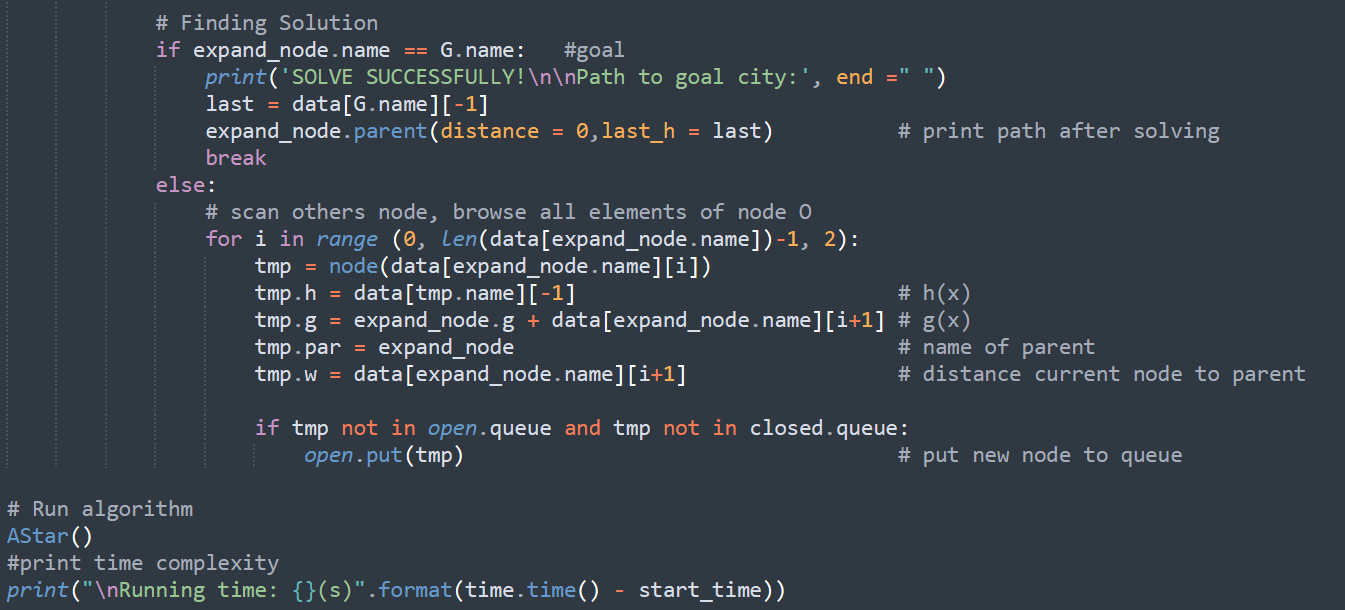
2.3.Algorithm optimization and synchronization

We use Github to optimize storage and code updates, Teams, and Whiteboards to exchange ideas and work.

**3. Algorithms**

**A\***





- *Completeness*: Not complete. It is only complete if the state space is finite, and we avoid repeated states and all costs are >ε.

- *Time complexity*: The number of nodes expanded is exponential in the depth of the solution (the shortest path) d: O(bd), where b is the branching factor (the average number of successors per state).

- *Space complexity*: O(bd), It keeps all the generated nodes in memory.

- *Optimality*: Expand node in frontier with best evaluation function score f(n):

+ f(n) = h(n) + g(n)

+ h(n) : heuristic estimate of cost to get from n to goal.

+ g(n) : cost to get from initial state to n.

* Optimal when h(n) is admissible.

**Deep First Search:**

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

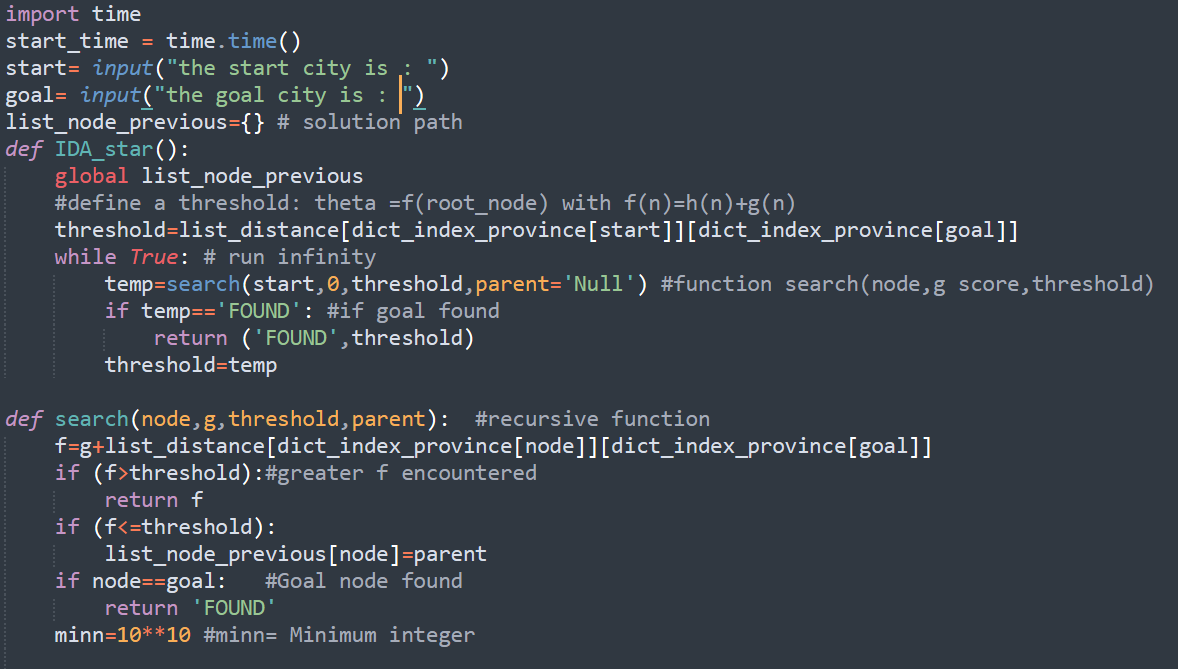
*- Completeness*: Not complete. But,it is complete in finite search spaces

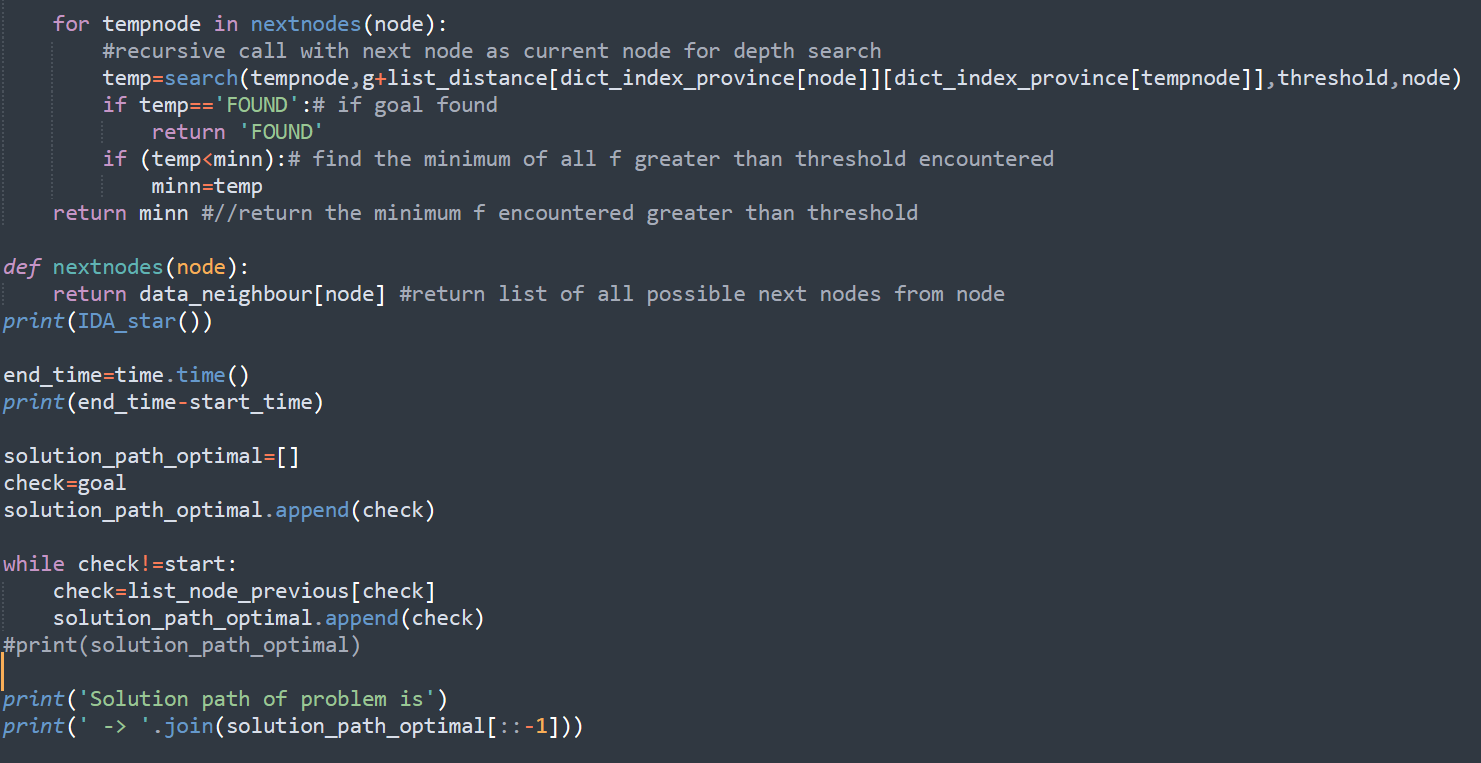
- *Time complexity*: O(bm), where b is the branching factor (the average number of successors per state) and m is maximum depth of the state space.

- *Space complexity*: Similar to time complexity, it is also O(bm), keep all nodes in memory.

- *Optimality*: Not optimal

**IDA\***





-*Completeness and optimal*: It is only complete if : h is admissible and , Finite branching factor

- *Time complexity*: The number of nodes expanded is exponential in the depth of the solution (the shortest path) d: O( bd ), where b is the branching factor (the average number of successors per state).

- *Space complexity*: O(bd) in the worst case.

**Uniform cost search:**



- *Completeness*: Yes, if step cost .

- *Time complexity*: O ( where C\* is the cost of the optimal solution.

- *Space complexity*: O (.

- *Optimality*: Yes - nodes expanded in increasing order of g(n).

**4.Comparing the result of the algorithm used for solving the problem:**

**4.1. Random seed data for accuracy**

With the data of 63 provinces, we used the function random.seed(42) to randomize the main warehouse as Son La, Nghe An and Ho Chi Minh, with the main warehouse list taking the first 10 elements of each domain ( seed(42)). Exporting to a CSV file, we have data of 300 interconnected components, and we compared each data line of 4 algorithms with the order of scoring 1,2,3 and 4 with min cost of 4. , max cost is 1. Below is code we used:

Graphical user interface, text

Description automatically generated

**4.2. Visualize data to compare algorithms**

Chart, pie chart

Description automatically generated

Chart, bar chart

Description automatically generated

**5. Conclusion and possible extensions:**

**5.1. Conclusion:**

The solution is proved by the optimization and completion of the procedure. Route planning is a problem with a wide variety of practical applications. Using four methods, we may choose the best solution to the issue in the most straightforward way. Furthermore, while evaluating and running our code, the A\* search approach is the most optimal when compared to IDA\* and A\* search.

**5.2. Possible extensions:**

Find a way to transport n goods through the warehouses for the most optimal cost and time.

**6. List of tasks:**

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Name | Contribution rate | |
| **SLIDE** | | | |
| Idea and slide 2,6,8,9,10 | Tran Vuong Quoc Dat | | 50% |
| Visualize, draw slide 1,3,4,6,7 | Nguyen Truong Truong An | | 50% |
| **ALGORITHM AND PROGRAMING** | | | |
| A Star | | | |
|  | Tran Vuong Quoc Dat | | 50% |
|  | Le Duc Anh Tuan | | 50% |
| IDA | | | |
|  | Nguyen Nho Trung | | 60% |
|  | Nguyen Thanh Dat | | 40% |
| UCS | | | |
|  | Tran Vuong Quoc Dat | | 60% |
|  | Nguyen Truong Truong An | | 40% |
| DFS | | | |
|  | Nguyen Truong Truong An | | 50% |
|  | Nguyen Thanh Dat | | 50% |
| Problem modeling | | | |
|  | Tran Vuong Quoc Dat | | 100% |
| Program Presentation | | | |
| Add comment, visualize and arrange programing | Le Duc Anh Tuan | | 100% |
| **REPORT** | | | |
| Introduction and modeling( 1,2) | Tran Vuong Quoc Dat | | 30% |
| Algorithm analysis(3) | Nguyen Thanh Dat | | 35% |
| Result visualization and conclusion(4,5) | Le Duc Anh Tuan | | 35% |
| **VIDEO** | | | |
|  | Nguyen Nho Trung | | 100% |
| **PROGRAMMING AND TASK SYNCHRONIZATION** | | | |
| Programing synchronization  (on Github) | Le Duc Anh Tuan | | 100% |
| Assign task and task synchronization | Tran Vuong Quoc Dat | | 100% |
| **DATA PROCESSING** | | | |
| Collect and create data files | Nguyen Nho Trung | | 40% |
| Create a formula that transforms data | Tran Vuong Quoc Dat | | 20% |
| Convert raw data into usage data | Le Duc Anh Tuan | | 40% |

**7. References**

**GitHub**: github.com/tuanlda78202/Logistics