



Faculty of Engineering and Technology
Electrical and Computer Engineering Department

ENCS3340

Artificial Intelligence

Course Project.1

Search Algorithms for Route Navigation

Student's Name: Lojain Abdalrazaq.

ID: 1190707.

Student's Name: Arwa Doha.

ID: 1190324.

Instructor's Name: Dr. Adnan Yahya.

Section: 1.

May 15, 2022

Abstract

The aim of this project is to implement a graph that contain nodes represent the cities in Palestine. And then finding the optimal path from the source city to destination city using several searching algorithms such as A*, Depth First Search(DFS), Breadth First Search(BFS), greedy, and any other searching algorithms implemented by any programming language. According to our ID student's numbers, our duty is to implement the BFS, DFS, and A* searching algorithms, and testing them to find the path from one city to another city in Palestine.

Program Implementation

■ Input Files:

First, we have divided the input data into two files. The first file is the **Cities.txt** which contains a list of the 20 cities in Palestine. While the second file is **Test.txt** that contains a matrix of size 20 cities, divided by using the hashtag (#). For each index in the file, it has divided into 3 parts as the following:

Aerial, Car, Walking

The Aerial distance represent the straight line distance between the source and destination, while the Car value represents the car driving distance, finally the walking, represents the Walking distance, Note that the unconnected cities have the car cost -1.

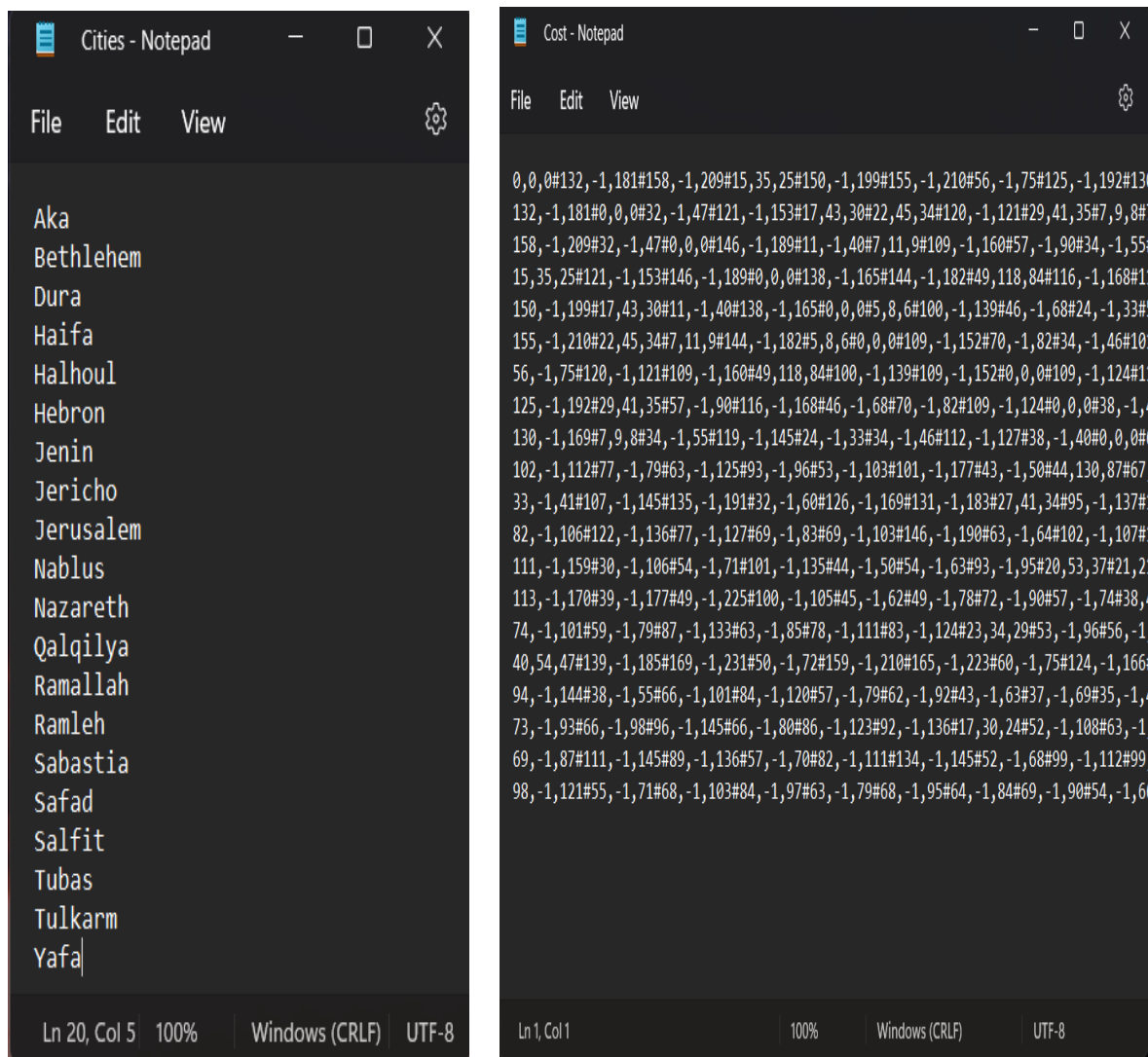


Figure 1 Input Files.

▪ Graph Implementation:

In our project, we have implemented the graph using the adjacency list using Java programming language, which means that we have a one dimensional array of the same size of the input cities represent each city, and each index in the array represent a linked list of the connected nodes together (have a car cost not equals to 0 or -1).

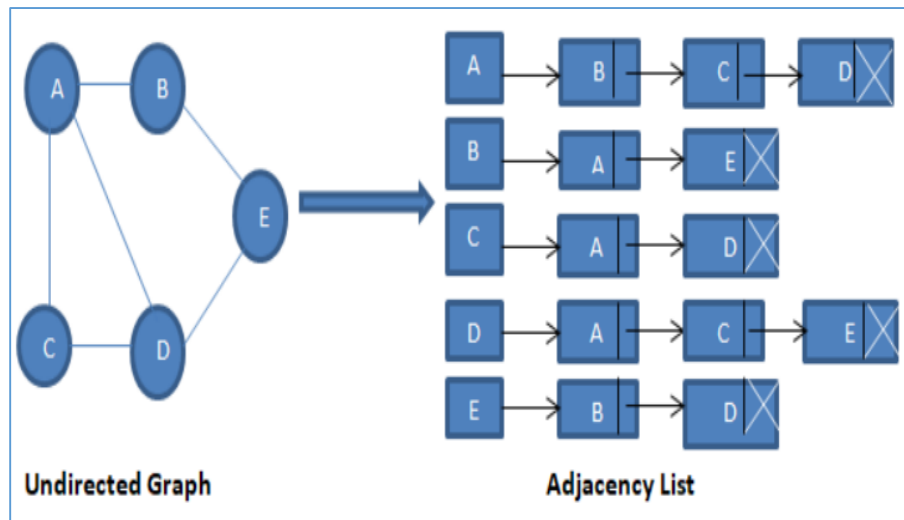


Figure 2 Graph Implementation.

In addition, we have created a class named **City**, this class contains a list of attributes such as , the source, destination, Aerial, Car, and finally a walking cost, in order to make it easier in reading the input files and storing the data (cities and cost) foe each two cities in Palestine.

```

/// The city class attributes.
/// The source city
/// considered as the start node
String Source;
/// The Destination city
/// considered as the goal node
String Destination;
/// The cost is the car distance between two connected cities or nodes
int car_cost;
/// The first heuristic from the source node to the destination
/// represent the walking distance
int h_walk;
/// The second heuristic from the source node to the destination
/// represent the straight line from the source node to the destination
int h_areal;

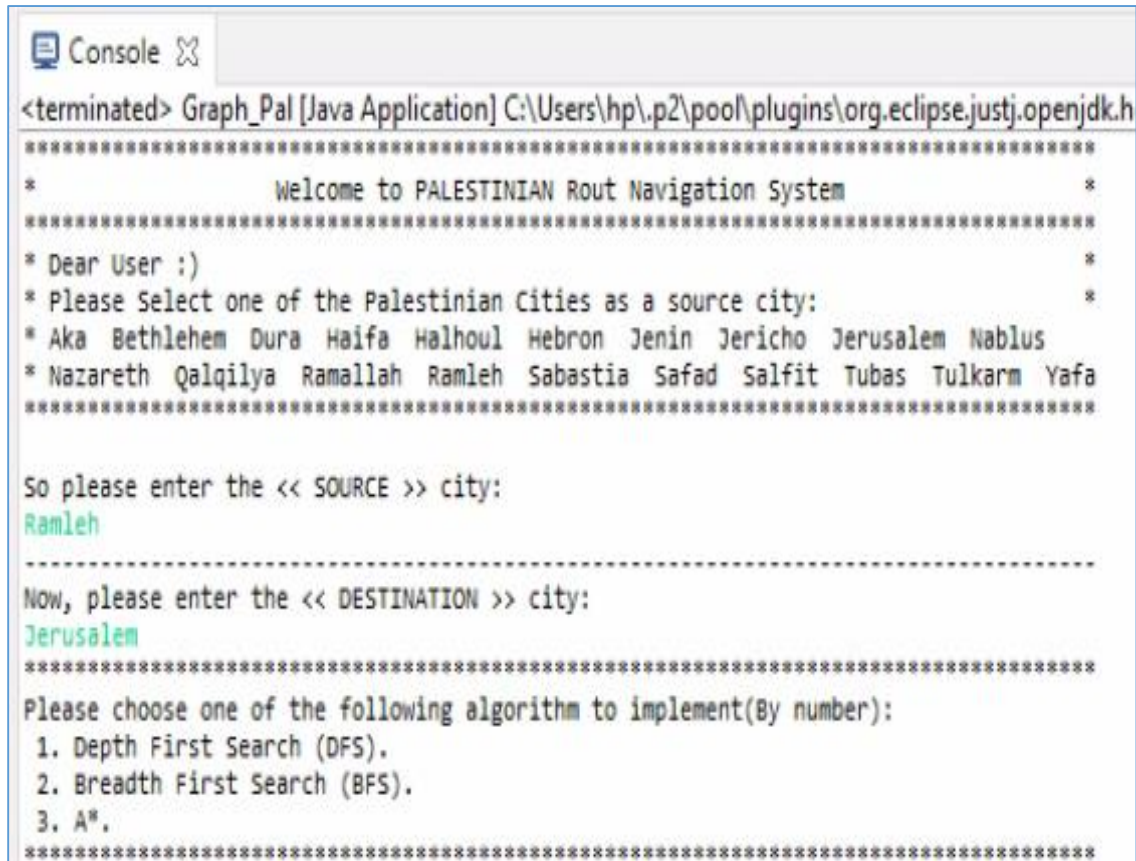
/// Initializing constructors

```

Figure 3 City class attributes.

▪ Programming Running:

We have implemented 3 searching algorithms (DFS, BFS, A*). In addition, the design of our system is to print all the Palestinian cities and let the user to choose the source and destination node. After that, he has to choose which algorithm to be implemented to find the path from the source and destination.



```
<terminated> Graph_Pal [Java Application] C:\Users\hp\.p2\pool\plugins\org.eclipse.justj.openjdk.h
*****
*                               Welcome to PALESTINIAN Rout Navigation System                               *
*****
* Dear User :)                                                         *
* Please Select one of the Palestinian Cities as a source city:         *
* Aka Bethlehem Dura Haifa Halhoul Hebron Jenin Jericho Jerusalem Nablus
* Nazareth Qalqilya Ramallah Ramleh Sabastia Safad Salfit Tubas Tulkarm Yafa
*****

So please enter the << SOURCE >> city:
Ramleh

-----
Now, please enter the << DESTINATION >> city:
Jerusalem
*****
Please choose one of the following algorithm to implement(By number):
1. Depth First Search (DFS).
2. Breadth First Search (BFS).
3. A*.
*****
```

Figure 4 The console output to the user.

✓ Depth First Search:

Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node and explores as far as possible along each branch before backtracking. So the basic idea is to start from the root or any arbitrary node and mark the node and move to the adjacent unmarked node and continue this loop until there is no unmarked adjacent node. So, to test the implementation in our program, we have chosen the Depth First Search to find the path from Ramleh to Jerusalem. **The output PATH is as the following:**

```
*****
So please enter the << SOURCE >> city:
Ramleh
-----
Now, please enter the << DESTINATION >> city:
Jerusalem
*****
Please choose one of the following algorithm to implement(By number):
1. Depth First Search (DFS).
2. Breadth First Search (BFS).
3. A*.
*****
1
-----<< Depth First Search (DFS) >>-----
The path from Ramleh to Jerusalem is as the following:
Ramleh Yafa Qalqilya Nablus Tubas Jenin Sabastia Tulkarm Nazareth Safad Aka Haifa Salfit Ramallah Jerusalem
*****
```

Figure 5 DFS output.

✓ Breadth First Search:

Breadth-first search (BFS) is an algorithm for searching a tree data structure for a node that satisfies a given property. It starts at the tree root and explores all nodes at the present depth prior to moving on to the nodes at the next depth level. Extra memory, usually a queue, is needed to keep track of the child nodes that were encountered but not yet explored. So, to test the implementation in our program, we have chosen the Breadth First Search to find the path from Bethlehem to Nazareth. **The output PATH is as the following:**

```
Console
<terminated> Graph_Pal [Java Application] C:\Users\hpl\p2\pools\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_14.0.2.v20200815-0932\jre\bin\java.exe
*****
Welcome to PALESTINIAN Rout Navigation System
*****
* Dear User :)
* Please Select one of the Palestinian Cities as a source city:
* Aka Bethlehem Dura Haifa Halhoul Hebron Jenin Jericho Jerusalem Nablus
* Nazareth Qalqilya Ramallah Ramleh Sabastia Safad Salfit Tubas Tulkarm Yafa
*****
So please enter the << SOURCE >> city:
Bethlehem
-----
Now, please enter the << DESTINATION >> city:
Nazareth
*****
Please choose one of the following algorithm to implement(By number):
1. Depth First Search (DFS).
2. Breadth First Search (BFS).
3. A*.
*****
2
-----<< Breadth First Search (BFS) >>-----
The path from Bethlehem to Nazareth is as the following:
Bethlehem Halhoul Hebron Jericho Jerusalem Dura Nablus Ramallah Ramleh Qalqilya Sabastia Salfit Tubas Yafa Jenin Tulkarm Haifa Nazareth
*****
THANK YOU!
*****
```

Figure 6 The BFS output.

✓ A*:

A* is a searching algorithm that searches for the shortest path between the initial and the final state. It is used in various applications, such as maps & graphs. A* algorithm is used to calculate the shortest distance between the source and the destination, and its estimated total cost of the path through node n to the goal.

Now, also to test the system, we have tried to find the Path and the total cost of the optimal solution (path) between Aka and Sabastia, and the output of the system is as the following:

```
Console
<terminated> Graph_Pal [Java Application] C:\Users\hp\.p2\pool\plugins\org.eclipse.justj.openjdk
*****
*                               Welcome to PALESTINIAN Rout Navigation System                               *
*****
* Dear User :)                                                         *
* Please Select one of the Palestinian Cities as a source city:         *
* Aka Bethlehem Dura Haifa Halhoul Hebron Jenin Jericho Jerusalem Nablus *
* Nazareth Qalqilya Ramallah Ramleh Sabastia Safad Salfit Tubas Tulkarm Yafa *
*****

So please enter the << SOURCE >> city:
Aka

-----
Now, please enter the << DESTINATION >> city:
Sabastia

Please choose one of the following algorithm to implement(By number):
1. Depth First Search (DFS).
2. Breadth First Search (BFS).
3. A*.
3

-----<< A* Search Algorithm >>-----
The path from Aka to Sabastia is as the following:
Aka
Haifa
Safad
Jenin
Nazareth
The desired destination found *FINALLY*--> Sabastia
The final cost --> 187.0

*****
*                               THANK YOU!                               *
*****
```

Figure 7 The output of A*.