Faculty of Engineering and Technology
Department of Electrical and Computer Engineering

ENCS 3340
Artificial Intelligence

AI Programming Project 1

Search for Optimal Service

Prepared By:

Marah Anabtawi -1171326
Farah Abu Lebdeh -1171456

Instructor:
Dr. Adnan Yahya

Section: #1

Date:13/04/2021

1 ABSTRACT

This project aims to learn search algorithms by implementation this algorithms using high level language. It is about searching on the Palestinian map for cities to reach with optimal performance using three different algorithms (Depth First Search , A* Search , Optimal2-All goals). Several studies about shortest path search show the feasibility of using graphs for this purpose.

Table Of Contents

2 Problem Formalization

- **Problem Formulation:**

1. **Initial state:** The user choose the initial state after read data and run the algorithm.
2. **Goal State:** The user choose the goal state after choosing initial data.
3. **Successor Function:** Moving from one place to another to reach the target according to the algorithm chosen by the user
4. **Function Used :**
   - **A* Search:** A* is an informed search algorithm, or a best-first search, meaning that it is formulated in terms of weighted graphs: starting from a specific starting node of a graph, it aims to find a path to the given goal node having the smallest cost (least distance travelled, shortest time)
   - **Depth Search:** Depth-first search (DFS) is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node (selecting some arbitrary node as the root node in the case of a graph) and explores as far as possible along each branch before backtracking.
   - **Optimal2-All goals:** Deals with the time as the main parameter to be optimized and time is the function of the distance and congestion factor per edge: meaning at rush hour the time is max and at other times the time is less and requires that we visit all the goal nodes in the table.

3 Problem Solution

In our solution, we used java and scene builder to build the project. First, after read the requirements we started with an idea of what the interferes would look like.Then, we collected data about Palestinian cities, street distance (km), Areal distance (km) between them, and the X and Y coordinates on the map. Then we read the data file and drew the edge between each city and the other, and displayed it on the map. After that, we started writing the code for each algorithm, and we checked that the path diagram was correct.

3.1 A* Algorithm

We want to reach the target city from the starting city as quickly as possible. Here A* Search Algorithm comes to the rescue.What A* Search Algorithm does is that at each step it picks the node according to a value-'f' which is a parameter equal to the sum of two other parameters – 'g' and 'h'. We define 'g' and 'h' as simply as possible g = the movement cost to move from the starting point to a given point on the map, following the path generated to get there. h = the estimated movement cost to move from that given point on the map to the final destination. This is often referred to as the heuristic, which is nothing but a kind of smart guess. We really don't know the actual distance until we find the path, because all sorts of things can be in the way (walls, water, etc.).

This is the heuristic equation that we used:

h = max { abs(current_cell.x – goal.x) , abs(current_cell.y – goal.y) }

3.2 Depth-first search

Depth-first search is an algorithm for searching map . 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. Then backtrack and check for other unmarked nodes and traverse them. Finally print the nodes in the path.

3.3 Optimal2-All goals

In the beginning we built a function that calculates a random value in a certain range (0 to 2). This random value is called a factor for each city separately, and it is the congestion factor for each edge, if the value is less than 1 that means the road is free of congestion and we do not need additional time. But, if the parameter is greater than 1, this indicates that the road is congestion and we need additional time.

Then, in optimal2 we chose the A* Algorithm because A* Search algorithm is one of the best and popular technique used in path-finding and search mapping to find the optimal path when the heuristic is admissible

Algorithm	Complete?	Optimal?	Time complexity	Space complexity
BFS	Yes	If all step costs are equal	$O(b^d)$	$O(b^d)$
UCS	Yes	Yes	Number of nodes with $g(n) \leq C^*$	Number of nodes with $g(n) \leq C^*$
DFS	No	No	$O(b^m)$	$O(bm)$
A*	Yes	Yes	Number of nodes with $g(n)+h(n) \leq C^*$	Number of nodes with $g(n)+h(n) \leq C^*$

4 Additional Solutions

We have added additional algorithms to our program, such as Breadth First Search and the Uniform-Cost Search.

4.1 The Uniform-Cost Search

In this algorithm from the starting state we will visit the adjacent states and will choose the least costly state then we will choose the next least costly state from the all un-visited and adjacent states of the visited states, in this way we will try to reach the goal state (note we wont continue the path through a goal state), even if we reach the goal state we will continue searching for other possible paths(if there are multiple goals) . We will keep a priority queue which will give the least costliest next state from all the adjacent states of visited states .

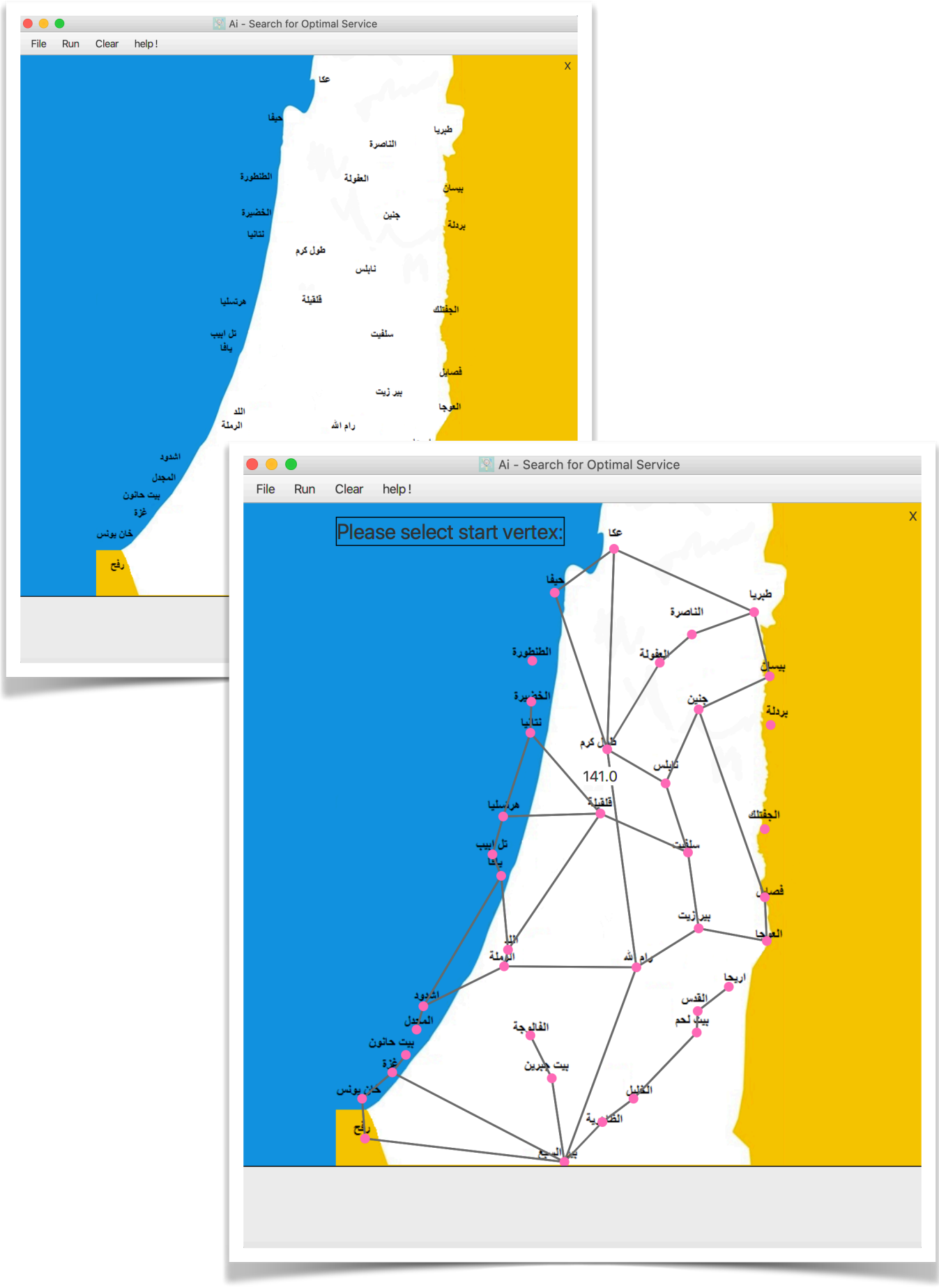
4.2 Breadth-first search (BFS)

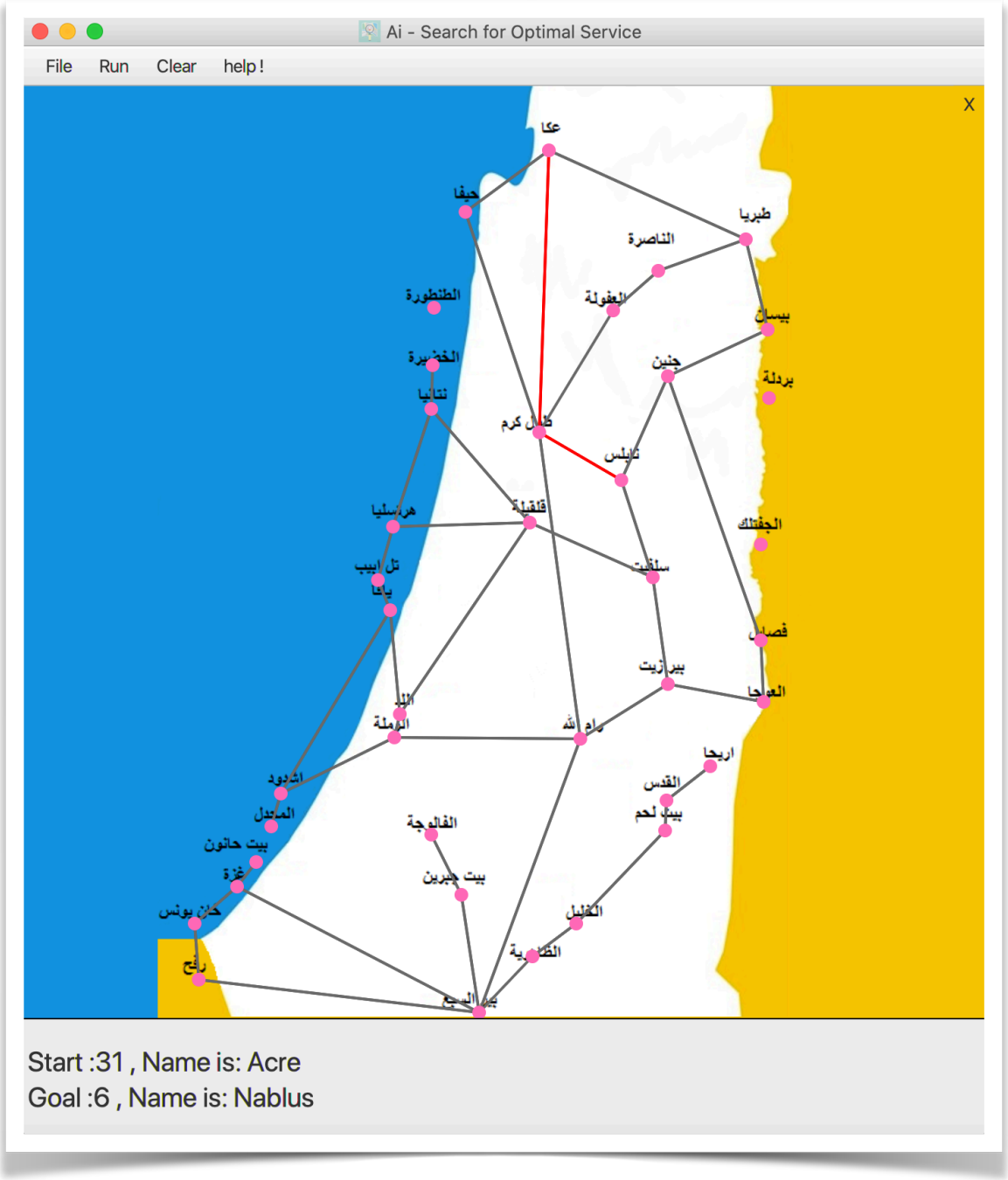
Breadth-first search (BFS) is an algorithm for traversing or searching map or graph data structures. It uses the opposite strategy of depth-first search, which instead explores the node branch as far as possible before being forced to backtrack and expand other nodes.

5 Conclusion

As a conclusion, Palestine mapping was solved and programmed using different algorithm and some of its heuristics, performing backtracking using JAVA language and the results was presented in a simple GUI. This project was completely challenging and informative but also very helpful as it helps in improving problem solving using Search for Optimal Service, programming skills and of course group work.

ScreenShots For Project:


Ai - Search for Optimal Service
File
Run
Clear
help!
Please select start vertex:
141.0
عكا
حيفا
طبريا
الناصرة
الطنطورة
العفولة
بيسان
الخضيرة
جنين
بردلة
نتانيا
طول كرم
نابلس
هرتسليا
قلقيلة
الجفتلك
تل ابيب
يافا
سلفيت
فصايل
بير زيت
العوجا
اللد
الرملة
رام الله
اشدود
المجدل
بيت حانون
غزة
خان يونس
رفح
اريحا
القدس
بيت لحم
الفالوجة
بيت جبرين
الخليل
الظاهرية
بير السبع


Ai - Search for Optimal Service
File
Run
Clear
help !
X
عكا
حيفا
طبريا
الناصرة
العفولة
الطنطورة
بيسان
الخضيرة
جنين
بردلة
نتانيا
كفار كرم
نابلس
هرتسليا
قلقيلية
الجفتلك
تل ابيب
سلفيت
يافا
فصايل
بيرزيت
العوجا
اللد
الرملة
رام الله
اريحا
اشدود
القدس
المجدل
بيت لحم
الفالوجة
بيت حانون
بيت جبرين
غزة
خان يونس
الخليل
رفح
الظاهرية
بير السبع
Start :31 , Name is: Acre
Goal :6 , Name is: Nablus