4511W Project Proposal

Cho, Chloe (cho00035) **Cho, Yongsuk (cho00085)** **Kim, Hyunwoo (kim00186)**

Description - Chloe

Maps use algorithms to find shortest or quickest path from the start location to the destination. We will be finding the shortest path in a map using various algorithms and comparing the performances of each one. The algorithms we will be considering are uninformed and informed graph searches. The search algorithms that we will use are A*, depth-first, depth-limited, iterative deepening, uniform cost, and bidirectional search. Another portion that we will explore is the A* heuristics where left turns are penalized since it takes longer. So we will add different A* algorithms with different heuristics to try and simulate an actual car driving on the road. This will not be a good representation of an actual car because it won't take traffic, car speed, or stop lights into consideration. All uninformed algorithms that will be used will run on the same map. This problem is interesting because we use smart navigation applications like Google Maps to help us find the quickest and shortest path to our destination. So being able to explore the different algorithms we learned in class and apply it to this project is something we are interested in. Also being able to see how different heuristics can affect the path found is something that will be interesting to explore.

Details - Chloe

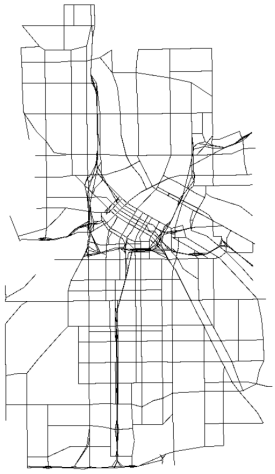

Figure 1: The result of drawing the map.csv file

To solve the problem we will use the map of Minneapolis shown in Figure 1 (map.csv) which is provided in the project idea description. The map has the x and y coordinate from the start of the segment and x and y coordinates for the end of the segment. There is also number 1 and 2 which represents the a one way street and two way street respectively. For the algorithms, we will modify the aima-python github code [1] to conform to the different parameters given in the map file. After all the code is implemented we will use python turtle graphics to show a visual representation of the map as well as the path that the algorithm finds as the shortest path. To compare the different data

of the search algorithms on the map, we will use the compare search function given in aima to help find the path cost. This will allow us to determine which search algorithm did best in finding the shortest path. The path length will be based off of the distance from the start coordinates to the end coordinates of the segment using the distance formula. We will also have the time of how long the algorithm takes to run the complete search to see which search algorithm is faster. For the best first search and A* search, a similar work has been done which can be seen in reference [4] to help find the shortest path. Using all the different resources, we can help solve the problem.

Results - Youngsuk

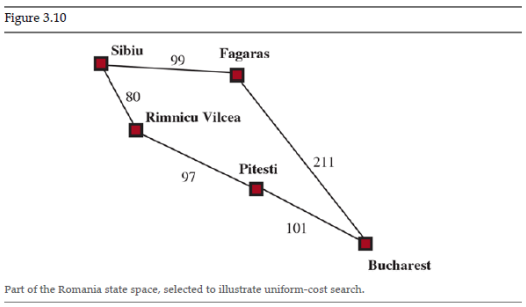

Figure 2: Weight Scale of the map

After running the different algorithms, the results that are obtained will be based of different factors. The research that is being conducted wants to compare each algorithm's characteristics with the results of the path's search time, how many nodes it has, and how much weight scale it has. As a result, the final efficiency model or algorithm's analysis will be taken through the differential treatment according to each outcome method. The overall result from the research will take the weight scale as the greatest score, how fast they get the search results, and how many nodes they have into consideration. The results form these three outcomes will be added to create a score that will rank each algorithm. This research will create a Minneapolis map through the given map.csv file and compare the search paths through each algorithm accordingly. The quality of the data requires a lot of search and results obtained from it, we believe that the Figure 2 can be seen as a preliminary result and predict the results between algorithms to some extent, but the actual and theory can be applied differently depending on each given condition. Following the reference data, A* and the uniform cost search consider the weight scale of each node. So based off the model, the results we expect are that the A* and uniform cost search will do the best [2].

Time Frame - Chloe

Rough indication but not finalized

- Short Literature Search Component due April 3
- Code Implementation due April 10
- Original Experimental Component due April 17
- Report Paper due April 30
- Final Project due May 3

Related Word - Hyunwoo

Search: "Before performing an action in a real world, the agent simulates the sequence of ACTIONS in the model and searches until it finds the sequence of actions that reach its GOAL. These sequences are called SOLUTIONS. The agent may need to simulate multiple sequences that have not reached its goal, but will eventually find the solution. Otherwise, you will see that the solution is impossible." [3].

We will be taking a look at both informed searches and uninformed searches. Informed search is when heuristics are used to help estimate the path. While uninformed search is where there is no given estimate and the path is searched blindly.

Uninformed Search Algorithms (Blind Search)

1. Breadth-first Search - Breadth-first search starts searching from the start node of the graph then expands all successor nodes at the current level before moving to nodes in the next level [2].

2. Depth-first Search - Depth-first search starts from the start node and goes through the path of the graph until the greatest depth node is reached before moving to the next path [2].

3. Depth-limited Search - A depth-limited search is similar to DFS but it had a predetermined limit. Depth limit will go until there are no successor nodes left [2].

4. Iterative Deepening Depth-First Search - The iterative deepening algorithm finds the best depth limit and does it by gradually increasing the limit until a goal is found [2].

5. Uniform Cost Search - The primary goal of the uniform-cost search is to find a path to the goal node which has the lowest cumulative cost. Uniform-cost search expands nodes according to their path costs from the root node [2].

6. Bidirectional Search – There are two simultaneous searches happening, one from the initial state which does a forward search and the other from the goal node which calls a backwards search. The search stops when these two graphs intersect each other. [2].

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening	Bidirectional (if applicable)
Complete?	Yes[1]	Yes[1,2]	No	No	Yes[1]	Yes[1,4]
Optimal cost?	Yes[3]	Yes	No	No	Yes[3]	Yes[3,4]
Time	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(b^m)$	$O(b^\ell)$	$O(b^d)$	$O(b^{d/2})$
Space	$O(b^d)$	$O(b^{1+\lfloor C^*/\epsilon \rfloor})$	$O(bm)$	$O(b\ell)$	$O(bd)$	$O(b^{d/2})$

Figure 3: Comparing Uninformed Search Algorithms

Informed Search Algorithms (Heuristic Search)

1. Best First Search (Greedy Search) - Greedy best-first search always selects the path that seem to be the best at the moment using the heuristic function and search. It expands the node that is closest to the goal node and the closest cost is estimated by heuristic function. This can be seen in Figure 2 [2].

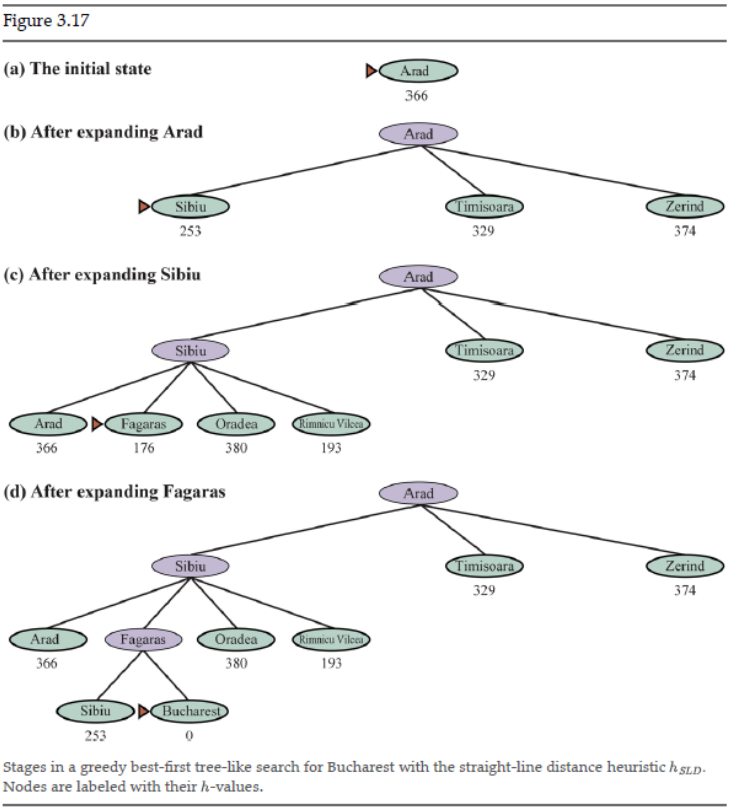

Figure 4: Greedy Best Search

2. A* Search Algorithm - A* search ranks to the shortest path through the search space using the heuristic function. This search algorithm expands less search tree and provides optimal result faster. A* algorithm is similar to uniform cost search except that it uses g(n) + h(n) which includes the heuristic. [4].

References

1. Norvig. *aima-python* `https://github.com/aimacode/aima-python`. (accessed: 03.10.2021).
2. J., M., L., R. & P., S. Comparative Analysis of Search Algorithms. *International Journal of Computer Applications* **179,** 40–43 (2018).
3. Russell, S. J. & Norvig, P. *Artificial Intelligence: a modern approach* 4th ed. (Pearson, 2020).
4. Potdar, G. P. & Thool, R. C. Comparison of Various Heuristic Search Techniques for Finding Shortest Path. *International Journal of Artificial Intelligence amp; Applications* **5,** 63–74 (2014).