NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

CZ3005 Artificial Intelligence (TEL4)

Assignment 1

Benjamin Ong (U2021759D)

Goh Shan Ying (U1921497C)

Isabelle Ng (U1922243C)

1. Introduction

This assignment focusing on finding the most efficient route of the New York city (NYC) road network of the United State within a given energy budget. This report will detail the implementation of the three different tasks as part of the CZ3005 assignment.

2. Task 1

2.1. Background

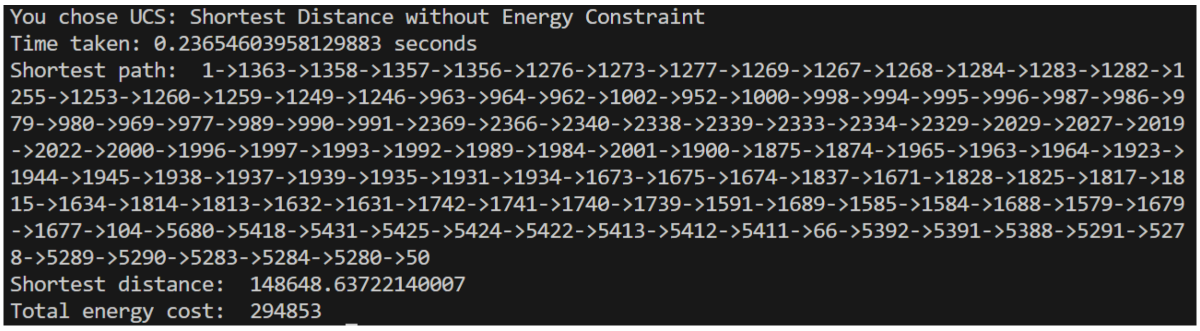
In Task 1, our baseline approach involves finding the shortest path without considering any energy constraints in the weighted graph where the distance are the weights.

2.2. Approach

Among the four algorithms available — BFS, DFS, UCS, and Dijkstra's — both BFS and DFS can be immediately ruled out. This is because they are not optimal for handling paths in weighted graphs. Between UCS and Dijkstra's algorithm, UCS is the preferred choice as the goal in this task is to find the distance from a single source node to a target node. Whereas in Dijkstra's algorithm, it enables us to find the shortest path from a single node to all other nodes [1]. Moreover, one key difference between the UCS and Dijkstra algorithms is their application. UCS is designed to handle large search problems for expansive graphs. On the other hand, Dijkstra's algorithm typically operates on a fully concrete graph [2]. Given that our focus is on identifying a path between two points within a huge graph, UCS emerges as the more suitable choice in this context.

In the baseline implementation of UCS we explore a graph by continuously selecting the unvisited path with the lowest cost. Starting from an initial node, UCS systematically expands the least costly path until it finds the target node. We can use a priority queue to ensure we always pick the least costly node to explore next.

2.3. Output



```
You chose UCS: Shortest Distance without Energy Constraint
Time taken: 0.23654603958129883 seconds
Shortest path:  1->1363->1358->1357->1356->1276->1273->1277->1269->1267->1268->1284->1283->1282->1
255->1253->1260->1259->1249->1246->963->964->962->1002->952->1000->998->994->995->996->987->986->9
79->980->969->977->989->990->991->2369->2366->2340->2338->2339->2333->2334->2329->2029->2027->2019
->2022->2000->1996->1997->1993->1992->1989->1984->2001->1900->1875->1874->1965->1963->1964->1923->
1944->1945->1938->1937->1939->1935->1931->1934->1673->1675->1674->1837->1671->1828->1825->1817->18
15->1634->1814->1813->1632->1631->1742->1741->1740->1739->1591->1689->1585->1584->1688->1579->1679
->1677->104->5680->5418->5431->5425->5424->5422->5413->5412->5411->66->5392->5391->5388->5291->527
8->5289->5290->5283->5284->5280->50
Shortest distance:  148648.63722140007
Total energy cost:  294853
```

Figure 1 Output of Task 1

The output displays the shortest path from Node 1 to Node 50. Given that UCS is both optimal and complete, it guarantees the path with the lowest cost. However, the energy expense exceeds the energy budget.

3. Task 2

3.1. Background

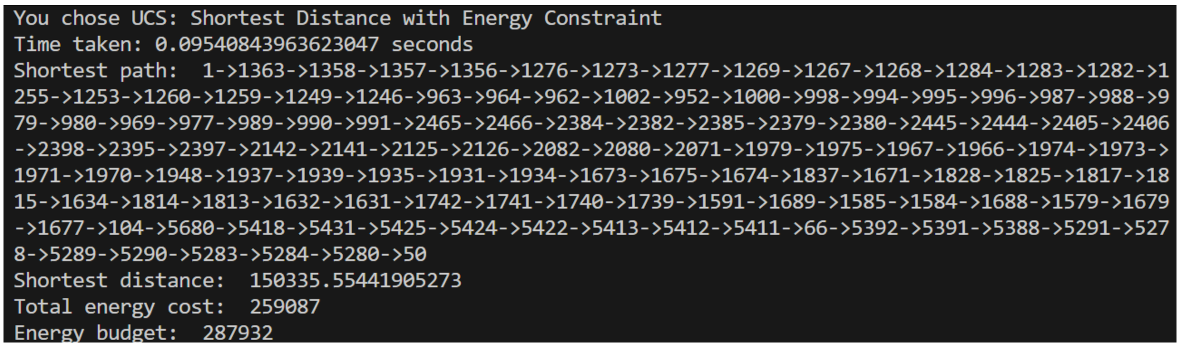
In Task 2, our initial method entails determining the shortest path in a weighted graph, where the distances serve as weights, without considering any energy limitations. We build upon the algorithm from Task 1 while considering energy constraints. Our refined method also incorporating dynamic weights that aim to strike a balance between minimizing energy cost and minimizing distance [3].

3.2. Approach

In our approach, we deal with energy constraints by dynamically adjusting the weights assigned to distance and cost during the Uniform Cost Search (UCS) algorithm. Initially, we set the weights at 0.7 for distance and 0.3 for cost, providing a balanced starting point for path exploration [3]. However, we recognize that in scenarios with energy constraints, ensuring that the total energy cost does not exceed the given budget is crucial. To achieve this, we conduct a trial-and-error process to fine-tune the weights. After experimentation, we find that setting the weight for distance to 0.9 and the weight for cost to 0.1 is the most effective approach.

By dynamically adjusting these weights, our modified UCS algorithm ensures that the total energy cost remains within the specified budget.

3.3. Output



```
You chose UCS: Shortest Distance with Energy Constraint
Time taken: 0.09540843963623047 seconds
Shortest path:  1->1363->1358->1357->1356->1276->1273->1277->1269->1267->1268->1284->1283->1282->1
255->1253->1260->1259->1249->1246->963->964->962->1002->952->1000->998->994->995->996->987->988->9
79->980->969->977->989->990->991->2465->2466->2384->2382->2385->2379->2380->2445->2444->2405->2406
->2398->2395->2397->2142->2141->2125->2126->2082->2080->2071->1979->1975->1967->1966->1974->1973->
1971->1970->1948->1937->1939->1935->1931->1934->1673->1675->1674->1837->1671->1828->1825->1817->18
15->1634->1814->1813->1632->1631->1742->1741->1740->1739->1591->1689->1585->1584->1688->1579->1679
->1677->104->5680->5418->5431->5425->5424->5422->5413->5412->5411->66->5392->5391->5388->5291->527
8->5289->5290->5283->5284->5280->50
Shortest distance:  150335.55441905273
Total energy cost:  259087
Energy budget:  287932
```

Figure 2 Output of Task 2

In Figure 2, the energy cost is successfully maintained below the budget constraint, with a total cost of 259087, which is less than the energy budget of 287932. This output provides a clear depiction of the shortest path from Node 1 to Node 50, highlighting the success of our approach in balancing energy efficiency and path optimization.

4. Task 3

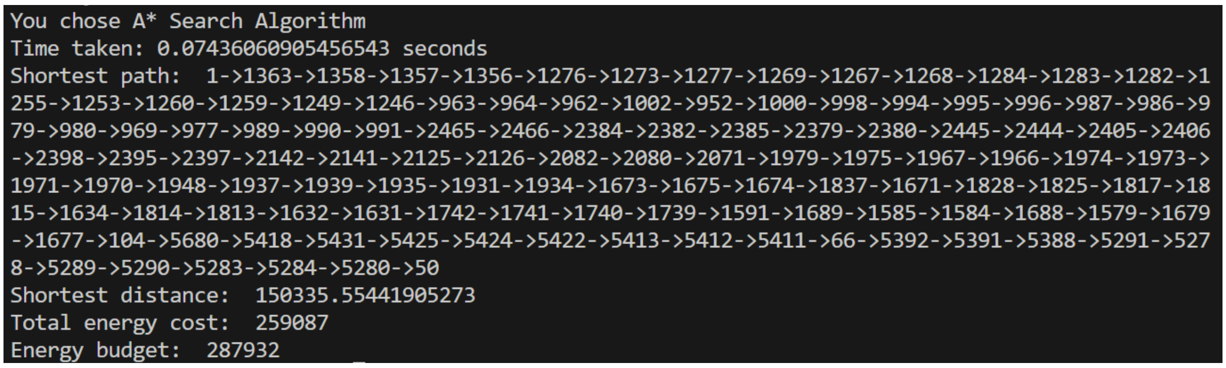
4.1. Background

In task 3, we are tasked to find the shortest path from a given source node to the destination node using A star search algorithm. Similar to task 2, we have an energy constraint.

4.2. Approach

The A star search algorithm is a refinement to the shortest path algorithm that directs the search towards the desired goal rather than exploring nodes based simply on distance from initial vertex [4]. The key idea is implementing a heuristic that estimates how far a given vertex v is from the goal vertex. When we pick a vertex to dequeue, rather than choosing current minimum distance, we choose vertex that minimizes the sum h(v) + dist(v) [4]. The heuristic of choice is the Euclidean distance.

4.3. Output



```
You chose A* Search Algorithm
Time taken: 0.07436060905456543 seconds
Shortest path:  1->1363->1358->1357->1356->1276->1273->1277->1269->1267->1268->1284->1283->1282->1
255->1253->1260->1259->1249->1246->963->964->962->1002->952->1000->998->994->995->996->987->986->9
79->980->969->977->989->990->991->2465->2466->2384->2382->2385->2379->2380->2445->2444->2405->2406
->2398->2395->2397->2142->2141->2125->2126->2082->2080->2071->1979->1975->1967->1966->1974->1973->
1971->1970->1948->1937->1939->1935->1931->1934->1673->1675->1674->1837->1671->1828->1825->1817->18
15->1634->1814->1813->1632->1631->1742->1741->1740->1739->1591->1689->1585->1584->1688->1579->1679
->1677->104->5680->5418->5431->5425->5424->5422->5413->5412->5411->66->5392->5391->5388->5291->527
8->5289->5290->5283->5284->5280->50
Shortest distance:  150335.55441905273
Total energy cost:  259087
Energy budget:  287932
```

The output displays the shortest path from Node 1 to Node 50, the shortest distance calculated, as well as the total energy cost. The result for A Star search is identical to the one from task 2, with the same path, energy cost and distance. However, the run time for the algorithm is slightly improved. While the algorithm was unable to find a more optimal path compared to UCS, the usage of a heuristic function made it more efficient and was able to get to the result faster.

5. Conclusion

Task Number	1	2	3
Distance (2 d.p)	148648.64	150335.55	150335.55
Energy Cost	294853	259087	259087
Time Taken (2 d.p)	0.237 secs	0.095 secs	0.074secs

In conclusion, for this set of data, A star algorithm seems to have the highest performance, reaching the result with the lowest time taken. While the usage of UCS in task 1 had a path with shorter distance, the A star algorithm was able to find a slightly longer path with much less energy cost in a shorter time due to the usage of a heuristic function.

While attempting the project, our group has learnt to apply the different algorithms and learnt to implement them ourselves. We also learnt to tweak our implementation according to the requirements and constraints of each task. While doing the project, we have learnt how changing the weights can affect the distance and cost in the result.

6. Contributions

Name	Contributions
Isabelle Ng	Code, Report
Goh Shan Ying	Code, Report
Benjamin Ong	Code, Report

All members contributed equally.

7. References

[1] C.S. (2019). Lecture 5: Search I.
https://web.stanford.edu/class/archive/cs/cs221/cs221.1196/lectures/search1.pdf

[2] GeeksforGeek (2023). Uniform Cost Search (Dijkstra for large graphs). Retrieved from https://www.geeksforgeeks.org/uniform-cost-search-dijkstra-for-large-graphs/

[3] I. Pohl, "The Avoidance of (Relative) Catastrophe, Heuristic Competence, Genuine Dynamic Weighting and Computational Issues in Heuristic Problem Solving," presented at Session 2 Theory of Heuristic Search, University of California, Santa Cruz, California.

[4] Cornell (2007) A* Search.

https://www.cs.cornell.edu/courses/cs312/2007sp/recitations/rec26.html#:~:text=An%20obvious%20choice%20for%20a,current%20node%20and%20the%20destination