Nanyang Technological University

CZ3005 Artificial Intelligence

Lab 1 Report

Team One

Group Member	**Matriculation Number**
David Tay Ang Peng	U1910603L
Joshua Toh Sheng Jie	U1921471F
Leong Hao Zhi	U1920469K

Submitted To - School of Computer Science and Engineering, Nanyang Technological University

Code Structure

For Lab 1, the team has chosen to implement the program using Python. We have chosen a modular structure, where each search algorithm has its own file. This enables higher flexibility and readability, as each class and algorithm has its own file. Having separate files for each search algorithm also enables faster and easier debugging of the various algorithms.

We also chose to use a class 'Data', which has parameters of start_node, end_node and energy_constraint. The class 'Data' loads the four data files as well as to store the starting node, ending node and energy constraint for different scenarios. This allows us to find the shortest path for different parameters without changing the script, but by instantiating a new Data class with different parameters.

Task 1

In Task 1, we were tasked to find the shortest path from '1' to '50' without an energy constraint using an algorithm taught in the lectures. As we have to find the shortest path, it means that the algorithm has to guarantee optimality. The following compares the various algorithms taught for the given problem, which has differing costs between nodes:

Breadth-First Search (BFS)

As the costs between edges are not uniform, BFS does not return the shortest path as optimality for BFS is only guaranteed given uniform costs between edges, thus BFS is not suitable for the task.

Uniform-Cost Search (UCS)

As Uniform-Cost Search considers the edge costs and expands to an unexpanded node with the least path cost, UCS can guarantee optimality (shortest path) for our scenario.

Depth-First Search (DFS)

DFS does not guarantee optimality (shortest path), thus DFS is not suitable for the task.

Depth-Limited Search (DLS)

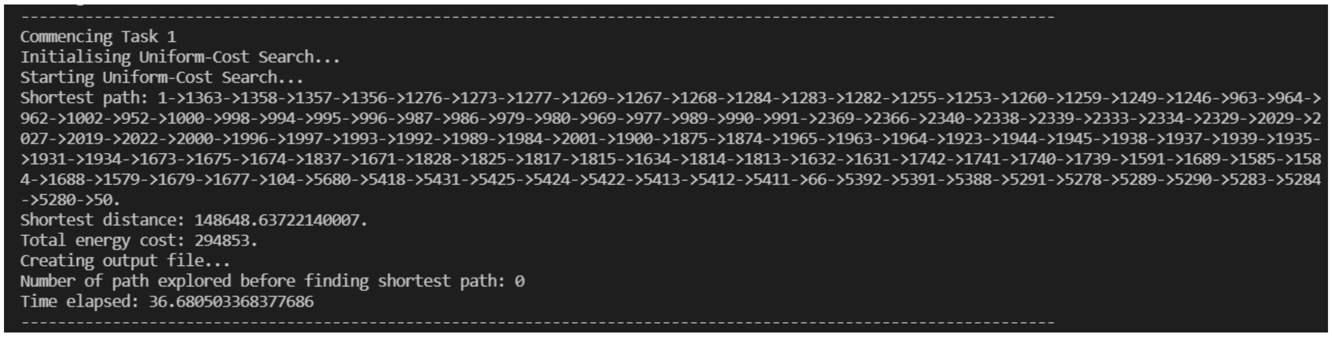
Similar to DFS, DLS does not guarantee optimality (shortest path), thus DLS is not suitable for the task.

Iterative Deepening Search (IDS)

As the costs between edges are not uniform, IDS does not return the shortest path as optimality for IDS is only guaranteed given uniform costs between edges, thus IDS is not suitable for the task.

Therefore, out of the five search algorithms taught in lecture, only Uniform-Cost Search guarantees optimality, or the shortest path in our scenario, given edges of differing costs. Therefore, we will use Uniform-Cost Search (UCS) for task 1.

After running the program, the following output which contains the path, distance and cost was obtained for task 1, which can be viewed in "Task1_ucs_without_energy_constraint_output.txt":



```
--------------------------------------------------------------------------------------------------------------
Commencing Task 1
Initialising Uniform-Cost Search...
Starting Uniform-Cost Search...
Shortest path: 1->1363->1358->1357->1356->1276->1273->1277->1269->1267->1268->1284->1283->1282->1255->1253->1260->1259->1249->1246->963->964->
962->1002->952->1000->998->994->995->996->987->986->979->980->969->977->989->990->991->2369->2366->2340->2338->2339->2333->2334->2329->2029->2
027->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->1815->1634->1814->1813->1632->1631->1742->1741->1740->1739->1591->1689->1585->158
4->1688->1579->1679->1677->104->5680->5418->5431->5425->5424->5422->5413->5412->5411->66->5392->5391->5388->5291->5278->5289->5290->5283->5284
->5280->50.
Shortest distance: 148648.63722140007.
Total energy cost: 294853.
Creating output file...
Number of path explored before finding shortest path: 0
Time elapsed: 36.680503368377686
--------------------------------------------------------------------------------------------------------------
```

Figure 1 - Shortest Path Using Uniform-Cost Search Without Energy Constraint

Task 2

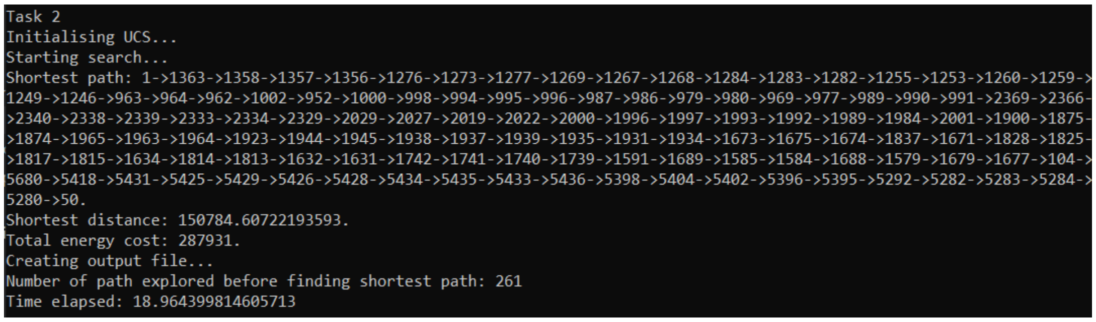
For task 2, with the additional constraint of an energy budget of 287932, our solution to task 1 using UCS does not meet this requirement.

With the intent of generating the optimal path, UCS is still the preferred algorithm. The additional feature of checking if the cost of the path found meets the energy budget requirements can be implemented by continuing the search with UCS until a path is found that meets the energy budget requirement of 287932.

However, this can be optimised by keeping track of the current cost of the path and only exploring neighbour nodes which cost does not result in exceeding the energy budget requirements. This reduces computation cost as paths that are found to not meet the energy budget requirements are not explored.

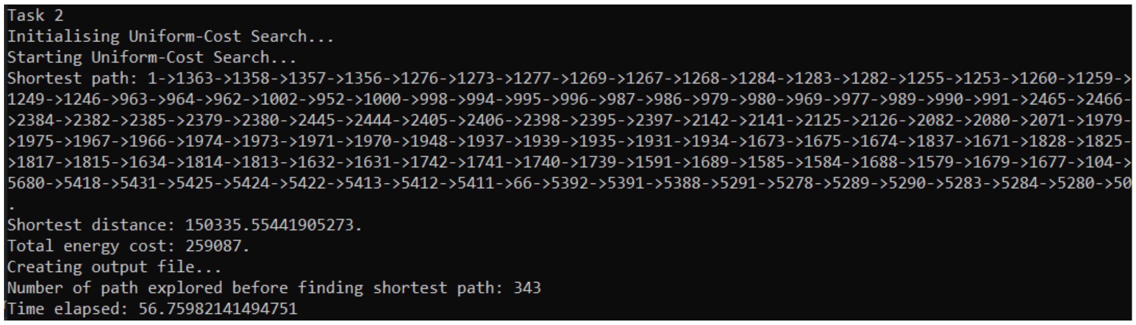
Additionally, as UCS aims to find the optimal path and explores the lowest cost nodes first, it can miss out on paths that are not the most optimal when there is no energy budget requirement, but would otherwise be the optimal path that adheres to the energy budget requirement. Thus, we need to explore all neighbour nodes that have a lower energy cost, regardless of the neighbour nodes' distance. This ensures the optimal path when an energy budget is present.

We compare the outputs of the program before and after this new implementation, as shown in figures 2 and 3, as well as “Task2_ucs_output.txt” and “Task2_ucs_optimised_output.txt”.



```
Task 2
Initialising UCS...
Starting search...
Shortest path: 1->1363->1358->1357->1356->1276->1273->1277->1269->1267->1268->1284->1283->1282->1255->1253->1260->1259->
1249->1246->963->964->962->1002->952->1000->998->994->995->996->987->986->979->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->1815->1634->1814->1813->1632->1631->1742->1741->1740->1739->1591->1689->1585->1584->1688->1579->1679->1677->104->
5680->5418->5431->5425->5429->5426->5428->5434->5435->5433->5436->5398->5404->5402->5396->5395->5292->5282->5283->5284->
5280->50.
Shortest distance: 150784.60722193593.
Total energy cost: 287931.
Creating output file...
Number of path explored before finding shortest path: 261
Time elapsed: 18.964399814605713
```

Figure 2 - UCS without exploring lower energy cost neighbours



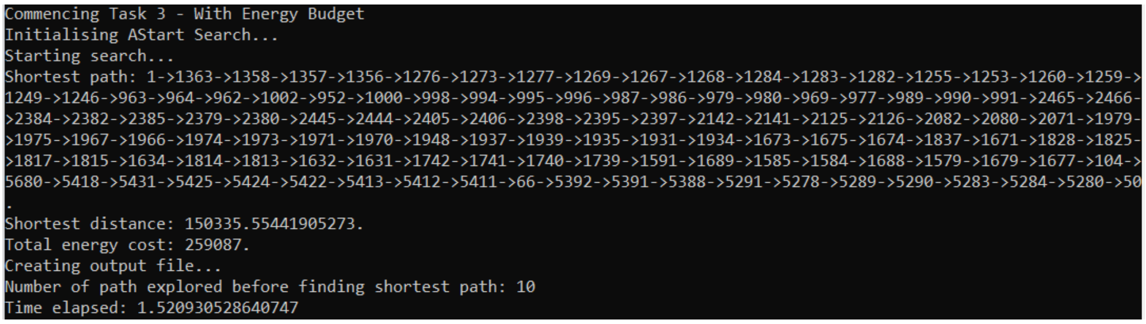
```
Task 2
Initialising Uniform-Cost Search...
Starting Uniform-Cost Search...
Shortest path: 1->1363->1358->1357->1356->1276->1273->1277->1269->1267->1268->1284->1283->1282->1255->1253->1260->1259->
1249->1246->963->964->962->1002->952->1000->998->994->995->996->987->986->979->980->969->977->989->990->991->2465->2466-
>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->1815->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->5278->5289->5290->5283->5284->5280->50
.
Shortest distance: 150335.55441905273.
Total energy cost: 259087.
Creating output file...
Number of path explored before finding shortest path: 343
Time elapsed: 56.75982141494751
```

Figure 3 - UCS while exploring lower energy cost neighbours

Task 3

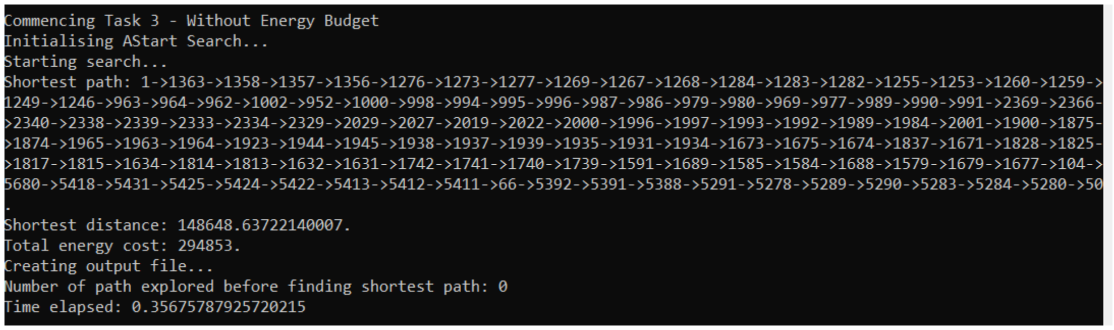
As for the last task, we need to decide on a heuristic function to use for the A* search algorithm. One of the most popular heuristics is the Euclidean distance. This heuristic will calculate the shortest distance between two nodes. It works in this case because any direction of movement is allowed when traversing the different locations in the map.

Compared to task 2, we can see that there is a lower total energy cost and significantly lower number of paths explored before finding the shortest path. This is due to the usage of a heuristic function which provides a shortcut to finding the solution.



```
Commencing Task 3 - With Energy Budget
Initialising AStart Search...
Starting search...
Shortest path: 1->1363->1358->1357->1356->1276->1273->1277->1269->1267->1268->1284->1283->1282->1255->1253->1260->1259->
1249->1246->963->964->962->1002->952->1000->998->994->995->996->987->986->979->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->1815->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->5278->5289->5290->5283->5284->5280->50
.
Shortest distance: 150335.55441905273.
Total energy cost: 259087.
Creating output file...
Number of path explored before finding shortest path: 10
Time elapsed: 1.520930528640747
```

Figure 4 - Shortest Path Using A* Search With Energy Constraint



```
Commencing Task 3 - Without Energy Budget
Initialising AStart Search...
Starting search...
Shortest path: 1->1363->1358->1357->1356->1276->1273->1277->1269->1267->1268->1284->1283->1282->1255->1253->1260->1259->
1249->1246->963->964->962->1002->952->1000->998->994->995->996->987->986->979->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->1815->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->5278->5289->5290->5283->5284->5280->50
.
Shortest distance: 148648.63722140007.
Total energy cost: 294853.
Creating output file...
Number of path explored before finding shortest path: 0
Time elapsed: 0.35675787925720215
```

Figure 5 - Shortest Path Using A* Search Without Energy Constraint

Conclusion

With different approaches used for each of the three tasks on the same problem, we explored and compared the efficiency of an uninformed search algorithm in Uniform Cost Search against an informed search algorithm by implementing a heuristic function.

Another factor to consider when choosing an algorithm is the efficiency of the algorithm. In our attempts, UCS explored 343 paths before finding the solution while A* Search explored 10 paths. This striking difference may not be important in this project as time is not a constraint, but the difference may be very costly in real-life situations where the speed of the algorithm is crucial. Therefore, one learning point we picked up was that accuracy may not be the only consideration we take when choosing an algorithm, as factors like efficiency do matter too.

As seen from the difference between task 2 and 3, the efficiency of the algorithm can be greatly improved by using a heuristic function. However, a heuristic function must satisfy 2 conditions:

1. Admissible	Heuristic never overestimates the true cost to the nearest goal.
2. Consistent	The heuristic step cost never overestimates the actual step cost.

Thus, we learnt that a good heuristic function can greatly improve the performance of an algorithm without sacrificing the accuracy too much, but also that having a fancy heuristic that improves speed tremendously may not necessarily produce accurate results.

Team Contribution

The team initially brainstormed together for ideas on how to tackle the various tasks together and came up with different approaches to the problems. Then, the team split up for the implementation of the different tasks as such:

- Task 1 - David Tay
- Task 2 - Leong Hao Zhi
- Task 3 - Joshua Toh

References

Soularidis, A. (2022, February 25). Uniform cost search (UCS) algorithm in Python. Medium. Retrieved March 10, 2022, from https://python.plainenglish.io/uniform-cost-search-ucs-algorithm-in-python-ec3ee03fca9f

Uniform-cost search (dijkstra for large graphs). GeeksforGeeks. (2021, August 25). Retrieved March 10, 2022, from https://www.geeksforgeeks.org/uniform-cost-search-dijkstra-for-large-graphs/https://towardsdatascience.com/a-star-a-search-algorithm-eb495fb156bb

Roy, B. (2021, April 26). *A-star (A*) search algorithm*. Medium. Retrieved March 10, 2022, from https://towardsdatascience.com/a-star-a-search-algorithm-eb495fb156bb