NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

CZ3005 (Artificial Intelligence)

Lab Assignment 1 Submission

By

AlphaBlur

Team Member Name	Matriculation No.
Koh Jia Cheng	U2022450J
Muqaffa Al-Afham Bin Kamaruzaman	U2022554C

Lab Group: TS4

Date of Submission: 26th Feb 2022

1. TASKS

1.1. Task 1 – NYC Instance without Energy Constraint

In task 1, the aim is to simply find the shortest path between the start (node 1) and end (node 50) nodes. Among the search algorithms taught in CX3005, breadth-first search (BFS), depth-first search (DFS), depth-limited-search (DLS) and greedy search can be ruled out as they do not guarantee an optimal solution, which is required by the problem statement. From the available options, uniform-cost-search (UCS) was chosen as it is the simplest to implement, and thus most likely least prone to errors and bugs. The following is the output achieved for task 1:

```
Running Task 1
================================================================================================================
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 ->    988 ->    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: 148648.637221
Total energy cost: 294853

[Additional info]
Total nodes processed: 5305
Max number of nodes in priority queue: 106
```

Note that the energy cost of the shortest path does exceed the given energy budget, thus it is expected that tasks 2 and 3 would give different paths.

1.2. Task 2 – Energy-Constrained NYC Instance (Uninformed Search)

In task 2, the aim is to find the shortest path whose total energy cost does not exceed the given energy budget, 287932.

UCS was again chosen for this task due to its simplicity. However, a few tweaks are needed for the algorithm to satisfy the energy constraint:

i) Each node is now associated with an energy cost.

ii) The priority queue may hold nodes with the same ID (these nodes are *peers*), but they will differ in distance from the start node, as well as energy cost.

iii) Priority is based **solely** on distance from start node.

iv) When considering to enqueue a node to the priority queue and it has no peer in the priority queue, this node is checked against the **largest-distance peer** from the ***visited*** set instead. E.g. Node 2 was already visited with distance-cost values = **(2, 12)**. Node 2 is again being considered but with values **(5, 10).** As 10 < 12, Node 2 is enqueued to the priority queue despite being 'visited'.

After applying these tweaks, the following output is achieved for task 2:

```
Running Task 2
================================================================================================
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: 150335.554419
Total energy cost: 259087

[Additional info]
Total nodes processed: 29009
Max number of nodes in priority queue: 595
```

The path is indeed different from the path from task 1, and the total energy cost is 259087, which is within the energy budget.

1.3. Task 3 – Energy-Constrained NYC Instance (A* Search)

In task 3, the aim is the same as task 2, but A* search must be used. The tweaks to UCS used in task 2 can also be applied to the algorithm for this task.

The only major change needed is the addition of a heuristic function *h(n)*, which estimates the path cost from a node n to the end node. Since the xy-coordinates of all nodes are given, *h(n)* will simply be the straight-line distance between n and the end node. The following is the output achieved for task 3:

```
Running Task 3
=============================================================================================================
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: 150335.554419
Total energy cost: 259087

[Additional info]
Total nodes processed: 3084
Max number of nodes in priority queue: 627
```

Note that the path given is the same as in task 2, suggesting that the algorithm used is optimal, which in turn suggests that *h(n)* is **admissible**. A significant improvement in performance is also observed, when comparing the number of nodes processed (dequeued from the priority queue) between the two algorithms. UCS processed **29009** nodes, whereas A* processed only **3084** nodes, implying a reduction of **89.4%** in terms of node processing times (assuming A*'s extra time on heuristics is negligible, which is fair considering that calculating h(n) is O(1) time-complexity).

2. CONCLUSION

Through this assignment, our group has learnt that standard search algorithms such as UCS can be modified to satisfy additional constraint(s). We have also learnt that by using well-defined heuristics functions, it is possible to significantly improve the performance of a search algorithm, without sacrificing the quality of the solution.

However, crafting an effective heuristic function is another challenge on its own. Although in our implementation we have used the straight-line distance between two coordinates already provided to us, there will be practical real-life problems where the heuristics function is not easily obtainable.

3. CONTRIBUTIONS OF TEAM MEMBERS

Task	Done By
Task 1	Koh Jia Cheng
Task 2	Koh Jia Cheng, Muqaffa Al-Afham Bin Kamaruzaman
Task 3	Muqaffa Al-Afham Bin Kamaruzaman
Report	Koh Jia Cheng, Muqaffa Al-Afham Bin Kamaruzaman

4. REFERENCES

(2012). Retrieved from Google Code: https://code.google.com/archive/p/json-simple/downloads/

Hasah, F. (n.d.). *What is uniform-cost search?* Retrieved from Educative: https://www.educative.io/edpresso/what-is-uniform-cost-search/

Shah, A. (2020, February 8). *How to Read JSON Object From File in Java?* Retrieved from Crunchify: https://crunchify.com/how-to-read-json-object-from-file-in-java/