Python / Algorithms, directed graphs

Task: Implement a (directed) graph data structure, as well as some searching algorithms to perform a state-space search on the graph. Before you choose which data structure to use to represent the graph (i.e., adjacency list, adjacency matrix, edge list, etc.), read through the whole assignment first, to determine which would be most suitable.

Random Graph Generation

Generate a random graph on n vertices, where $10 \le n \le 14$ is a randomly chosen integer.
At each vertex, generate $1 \le m \le 4$ distinct edges pointing to a randomly chosen vertex, such that:

- each edge is labelled with real-valued weight $0 \le w \le 1$,
- there are no multiple arcs $u \rightarrow v$ between the same ordered pair (u, v) of vertices in the graph (although loops are allowed).

Designate a random vertex as the initial vertex, and designate three distinct random vertices as target vertices.

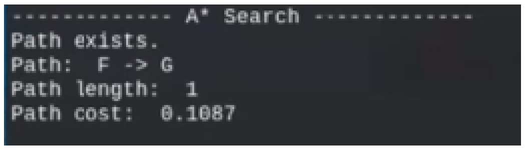
DOT file. Your program should create a DOT (.dot) file which will allow me to generate an image to visualise the graph. Make sure you indicate the initial and target vertices somehow.

Graph Searching

Implement the following algorithms to search the graph, starting from the initial vertex, to find one of the target vertices.

(a) Breadth-first search,	(d)	Iterative deepening search,
(b) Depth-first search,	(e)	A* search,
(c) Uniform-cost search,	(f)	Iterative deepening A* search.

Each algorithm should return and display a path (if one exists) from the initial vertex to one of the target vertices, together with the associated total cost of the path.



```
------------ A* Search ------------
Path exists.
Path:  F -> G
Path length:  1
Path cost:  0.1087
```

Warning. When implementing your algorithms, make sure to deal with cycles in your graph—they can cause a lot of problems.

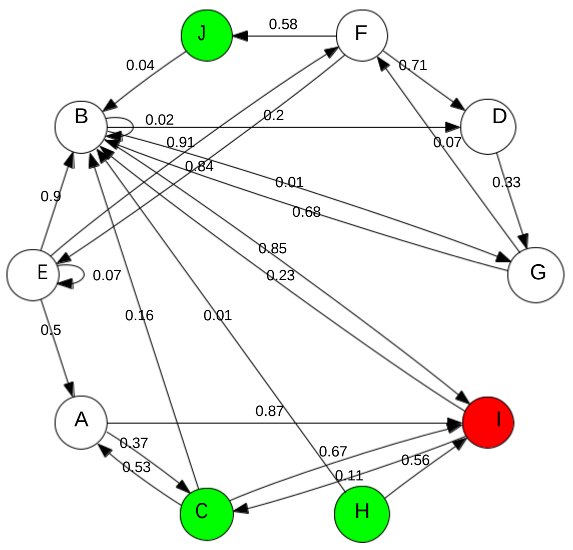


Figure 1: An example of a randomly generated graph, with the initial vertex in red, and target vertices in green. I created this image from a DOT file using the command line utility dot.

Behaviour of the Program

When I run the program, a random graph should be created, together with an associated DOT file, and each of the searching algorithms should be run on the graph. Your program should output n (the number of vertices), and the total number of edges in the graph. You should also output the path found using each algorithm, as well as the associated total cost of each path.

- Justify the graph data structure representation you chose.
- **Programming Language to be used Python 3**
- Packages allowed numpy, mathplotlib, network
- Comment the code so it will be more understandable