ID	Algorithm	Explored states	Solution	Path cost	Time (seconds)
1	Breadth First Tree Search	Fagaras, Sibiu, Bucharest, Arad	Sibiu, Arad, Zerind	314	0.0002364
2	Breadth First Graph Search	Fagaras, Sibiu, Bucharest Arad	Sibiu, Arad, Zerind	314	0. 0001594
3	Depth First Graph Search	Fagaras, Bucharest, Giurgiu, Pitesti, Craiova, Drobeta, Lugoj, Timisoara, Arad, Zerind	Bucharest, Pitesti, Craivova, Drobeta, Mehadia, Lugoj, Timisoara, Arad, Zerind	1019	0. 0003030
4	Uniform Cost Search	Fagaras, Sibiu, Rimnicu, Bucharest, Arad Oradea, Pitesti, Urziceni, Giurgiu, Zerind	Sibiu, Arad, Zerind	314	0.0005806
5	A* Search	Fagaras, Sibiu, Arad, Oradea Zerind	Sibiu, Arad, Zerind	314	0.0006126

Note: I found that the Depth First Tree Search won't work with the current dataset (without modification to the looping logic)

1. **Breadth-First Tree Search (BFTS) and Breadth-First Graph Search (BFGS):**
   - These algorithms use a FIFO queue for node expansion. Each node at the frontier (i.e., unexplored nodes) is expanded in the order they are added. This ensures that nodes closer to the root are explored first, leading to a level-order search tree exploration.

Student#: 8116775
Name: Michael McMillan

- In the data, both BFTS and BFGS found a path from Sibiu to Zerind with a total cost of 314 and completed the search within 0.0002364 and 0.0001594 seconds, respectively.

2. **Depth-First Graph Search (DFGS):**
   - This algorithm uses a LIFO stack for node expansion, meaning that the recently added node is first expanded. This results in deep exploration along a path before backtracking, leading to inefficient paths or revisiting nodes if loops are not appropriately handled.
   - According to the results, DFGS explored a much longer path, from Bucharest to Zerind, with a total path cost of 1019, taking 0.0003030 seconds. The long path reflects how depth-first search tends to go down deep paths before finding a solution.
3. **Uniform Cost Search (UCS):**
   - UCS uses a priority queue based on path cost. Nodes with the lowest cumulative cost are expanded first, making UCS optimal in terms of path cost when costs are non-negative.
   - UCS found the optimal path from Sibiu to Zerind with a cost of 314, in 0.0005806 seconds. This is slower than the BFS algorithms but guarantees finding the least-cost solution.
4. **A* Search:**
   - A* also uses a priority queue, but it expands nodes based on the sum of the path cost and a heuristic cost and a heuristic function estimating the cost to the goal. It balances between UCS and Greedy Search by considering both the path cost and a heuristic.
   - A* achieved the same path and cost of 314 as UCS and BFS but took 0.0006126 seconds, likely due to the additional overhead of evaluating the heuristic during each node expansion.

Performance Comparison

- **Efficiency:** Breadth-First Graph Search (BFGS) was the fastest algorithm, taking only 0.0001594 seconds. In contrast, A* was the slowest at 0.0006126 seconds.
- **Optimality:** BFGS, BFTS, UCS, and A* found the optimal path with a path cost of 314, whereas DFGS found a significantly longer path with a path cost of 1019.
- **LIFO vs FIFO:** DFGS, which uses a LIFO structure, results in more profound and potentially inefficient explorations, as seen in its more extended path cost. On the other hand, BFS algorithms (FIFO) are more systematic and optimal when costs are uniform.

Student#: 8116775
Name: Michael McMillan