Analysing Performance of Search Algorithms

I. Time analysis

The graph below illustrates the performance of the search algorithms concerning different graph configurations. Each algorithm's reaction to changes in graph density and size is evident, albeit with varying degrees.

a. Graph Density

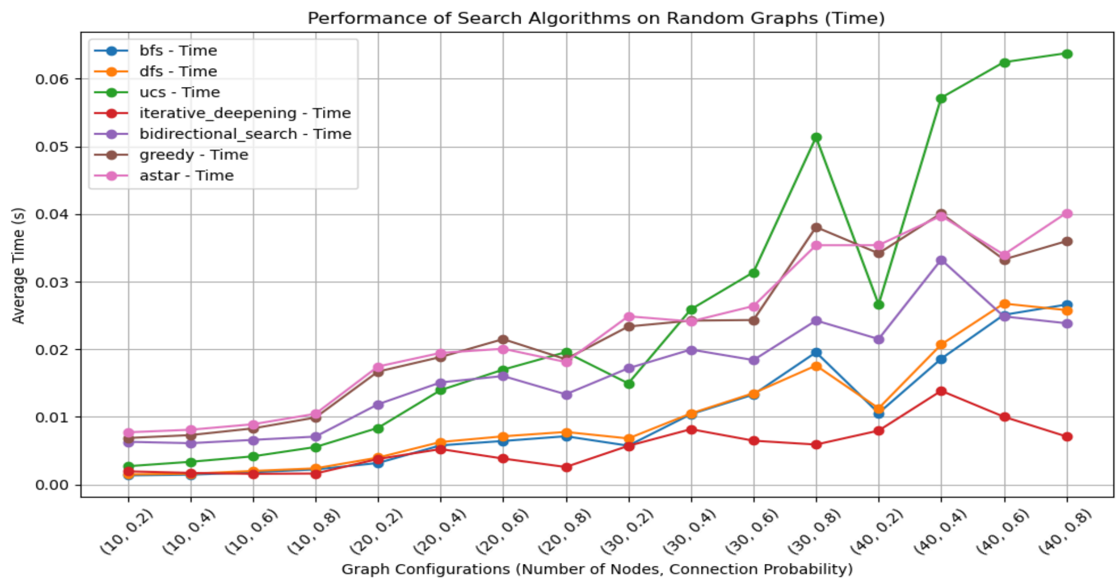
All algorithms exhibit a noticeable decrease in execution time as the graph becomes more connected, indicating improved efficiency in traversing denser graphs.
Among the algorithms, Iterative deepening demonstrates the most significant reduction in execution time as graph density increases, suggesting its effectiveness in navigating densely connected graphs efficiently.

b. Graph Size

As the number of nodes in the graph increases, the execution time of all algorithms tends to rise proportionally. This trend is particularly pronounced in UCS, where the increase in graph size leads to a more pronounced rise in execution time compared to other algorithms.
Notably, iterative deepening search maintains relatively consistent execution times across different graph sizes, indicating its robustness and efficiency in handling graphs of varying complexities.
UCS reacts with the highest degree of all algorithms.

c. Algorithm Comparisons

BFS and DFS exhibit similar performance characteristics across different graph configurations, highlighting their comparable efficiency in traversing graphs.
Greedy search and A* search also demonstrates similar execution times, indicating their analogous behaviour in finding optimal paths while utilising heuristic information.
Interestingly, iterative deepening search maintains a nearly constant execution time throughout the experiment, showcasing its stable performance regardless of changes in graph configurations.



II. Space analysis

The graph below illustrates the solution path length returned by the search algorithms concerning different graph configurations. Each algorithm's reaction to changes in graph density and size is evident, albeit with varying degrees.

a. Graph Density

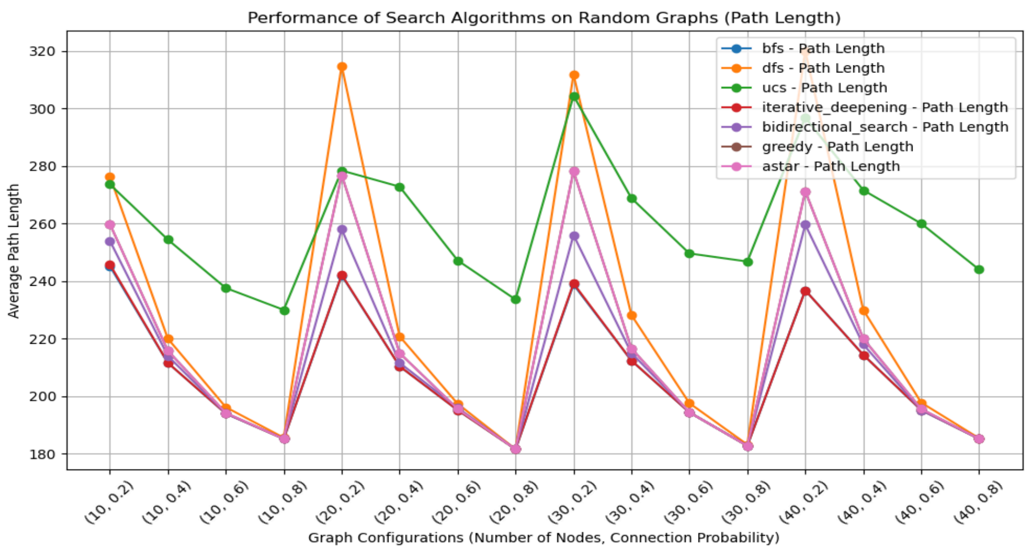
The graph below illustrates the solution path length returned by various search algorithms concerning different graph configurations. Notably, all algorithms exhibit a noticeable decrease in path length as the graph becomes more connected, indicating improved efficiency in traversing denser graphs. Among the algorithms, DFS demonstrates the most significant increment in path length as graph density increases, suggesting its ineffectiveness in navigating densely connected graphs efficiently.

b. Graph Size

Interestingly, graph size appears to have minimal impact on the path length returned by most algorithms, except for UCS. While the path length tends to increase slightly with larger graph sizes, the change is relatively small across all algorithms.
UCS stands out as the algorithm most affected by changes in graph size, with a noticeable increase in path length as the number of nodes in the graph increases. This suggests that UCS may struggle to maintain optimal path lengths in larger graph structures.

c. Algorithm Comparisons

UCS stands out as the outlier among the algorithms, exhibiting a distinct pattern in path length compared to the others. While the remaining algorithms generally follow similar trends in response to changes in graph density and size, UCS displays unique behaviour.
DFS reacts most negatively to increases in graph density, resulting in longer path lengths as the graph becomes less sparse. This highlights DFS's limitations in efficiently traversing densely connected graphs compared to other algorithms.
In contrast, the remaining algorithms exhibit relatively consistent behaviour in response to changes in graph density, with minor variations in path length.



Conclusion

In summary, the analysis of search algorithms' performance across different graph configurations reveals notable trends. Generally, denser graphs result in shorter execution times and path lengths for most algorithms, indicating their efficiency in traversing more connected graphs. Iterative deepening search stands out for its consistent performance across various graph densities, while UCS exhibits distinct behaviour with larger graph sizes. DFS shows limitations in traversing densely connected graphs efficiently. Overall, the findings emphasise the importance of considering graph characteristics when selecting search algorithms, with iterative deepening search demonstrating robustness across different graph complexities. Further investigation could provide deeper insights into algorithm behaviour and performance nuances.

Analysis of Centrality Measures in the Romania Cities Road Network

In our study of the road network graph representing cities in Romania, we conducted an analysis of various centrality measures to discern the importance and connectivity of different cities. Below, we present a formal analysis based on the findings obtained from applying centrality measures to the graph:

Cities	Degree	Closeness	Eigenvector	Katz	PageRank	Betweenness	Overall
Oradea	Rank 3						
Zerind	Rank 3						
Arad	Rank 2						
Timișoara	Rank 3						
Lugoj	Rank 3						
Mehadia	Rank 3						
Drobeta	Rank 3						
Craiova	Rank 2						
Sibiu	Rank 1	Rank 3	Rank 2	Rank 1	Rank 2	Rank 2	Rank 1
Rimnicu-Vilcea	Rank 2		Rank 1	Rank 3	Rank 3	Rank 3	Rank 3
Fagaras	Rank 3						
Pitesti	Rank 2	Rank 1	Rank 3	Rank 2			
Giurgiu							
Bucharest	Rank 1	Rank 2			Rank 1	Rank 1	Rank 2
Urziceni	Rank 2						
Eforie							
Harsova	Rank 3						
Vaslui	Rank 3						
Iasi	Rank 3						
Neamt							

Note: If a cell is empty it means the rank is below three.

Summery

Sibiu and Bucharest emerged as the cities with the highest degree-centrality. This indicates that both cities are pivotal nodes within the road network, boasting a significant number of connections to other cities. As such, they play a crucial role in facilitating travel and transportation across the region.

Pitesti exhibited the highest closeness centrality among all cities. This implies that Pitesti is strategically positioned within the road network, offering the shortest travel distances or times to reach other cities. Consequently, Pitesti serves as a central hub for efficient transportation routes throughout the region.

Riminicu Vilcea emerged as the city with the highest eigenvector centrality. This indicates that Riminicu Vilcea is closely connected to other cities that are themselves highly central within the road network. As a result, Riminicu Vilcea serves as a significant intermediary node, facilitating connections to other central cities.

Sibiu obtained the highest Katz centrality score, signifying its importance within the road network. Katz centrality considers both the number of immediate connections a city has and the significance of those connections. Sibiu's high Katz centrality suggests that it possesses numerous vital connections, both directly and indirectly, making it a crucial node within the network.

Bucharest attained the highest PageRank score, indicating its prominence and popularity within the road network. Analogous to Google's PageRank algorithm for web pages, Bucharest's high PageRank score suggests that it has numerous high-quality connections to other cities. Thus, Bucharest serves as a central destination with substantial influence and connectivity.

Bucharest demonstrated the highest betweenness-centrality among all cities. This highlights Bucharest's pivotal role in the road network, as it frequently lies along the shortest paths between other cities. As the capital city of Romania, Bucharest serves as a crucial junction and transit point, facilitating travel and connectivity throughout the region.

Finally overall Sibiu has highest centrality which indicates that this city holds a position of exceptional importance and influence within the network

In summary, our analysis of centrality measures in the Romania cities road network provides valuable insights into the importance and connectivity of different cities. By examining various centrality metrics, we gain a comprehensive understanding of the network's structure and the roles played by individual cities in facilitating transportation and connectivity across the region.

Comparing different graph representations

Note: V is number of vertex and E is number of edges

Time complexity

Operations	Adjacency list	Adjacency matrix	Edge list
Insertion of vertices	O(1)	O(1)	O(1)
Insertion of edges	O(1)	O(1)	O(1)
Deletion of vertices	O(V + E)	$O(V^2)$	O(V + E)
Deletion of edges	O(V)	O(1)	O(E)
Checking existence of an edge	O(V)	O(1)	O(E)
Finding neighbours of a vertex	O(degree(v))	O(V)	O(E)

Space complexity

Representation	Space Complexity
Adjacency List	O(V + E)
Adjacency Matrix	O(V^2)
Edge List	O(E) (assuming vertices are already known)