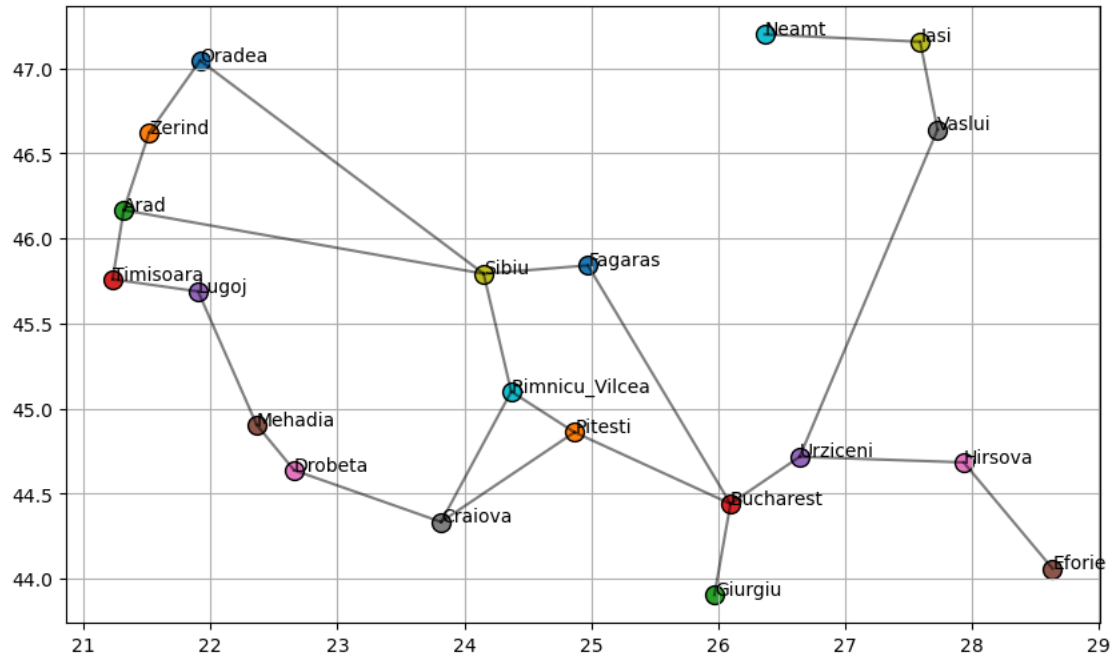


## Question 1:

### Introduction:

The plotted graph visualizes a geographical network of cities in Romania, showcasing the interconnectedness of urban centers within the region. Each node in the graph represents a distinct city, while edges between nodes denote direct connections between those cities. This visualization provides valuable insights into the spatial layout of the cities. Through this representation, we can better understand the geographic distribution and connectivity patterns of the cities, laying the groundwork for further analysis and exploration of the urban network's dynamics and characteristics.

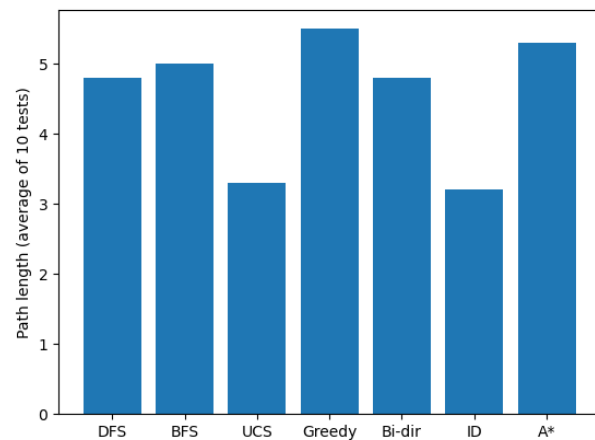
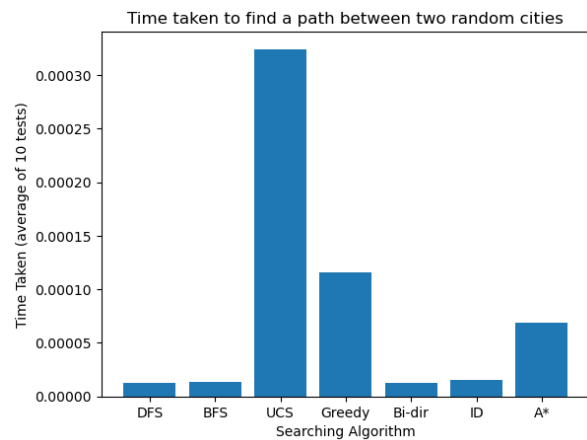


# Experiment 1: Analysis of Search Algorithms for Path-finding Efficiency

## Introduction:

In this experiment, we assess the efficacy of various search algorithms in finding paths between randomly chosen cities on a map. The algorithms including: DFS, BFS, UCS, Greedy Search, Bidirectional Search, ID, and A\*, each with its distinct approach to exploration. By measuring the time taken and the path length for 10 repetitions of each algorithm, we aim to gain insights into their performance characteristics.

Here are the graphs plotted visualizing the results of this experiment.



## Observations:

### Runtime (Time Taken):

- ❖ Uniform Cost Search (UCS) demonstrates the longest average runtime due to its exhaustive search based on node costs.
- ❖ Greedy Search and A\* Search follow UCS in runtime, utilizing heuristic information but requiring significant computational resources.
- ❖ BFS, Bidirectional Search, DFS, and ID exhibit comparatively lower average runtimes, known for their systematic exploration of the search space.

### Path Length:

- ❖ Greedy Search returns paths with the longest average length owing to its immediate gain prioritization.
- ❖ A\* Search produces paths of similar length to Greedy Search despite its informed nature.
- ❖ BFS yields shorter paths on average, reflecting its systematic exploration of all possible paths.
- ❖ Bidirectional Search and DFS demonstrate similar average path lengths, indicating comparable path optimality.
- ❖ UCS and Iterative Deepening return paths with the shortest average lengths, focusing on cost minimization and systematic deep exploration, respectively.

## Question 2:

### Experiment 2: Exploring Search Algorithm Performance in Randomly Generated Graphs

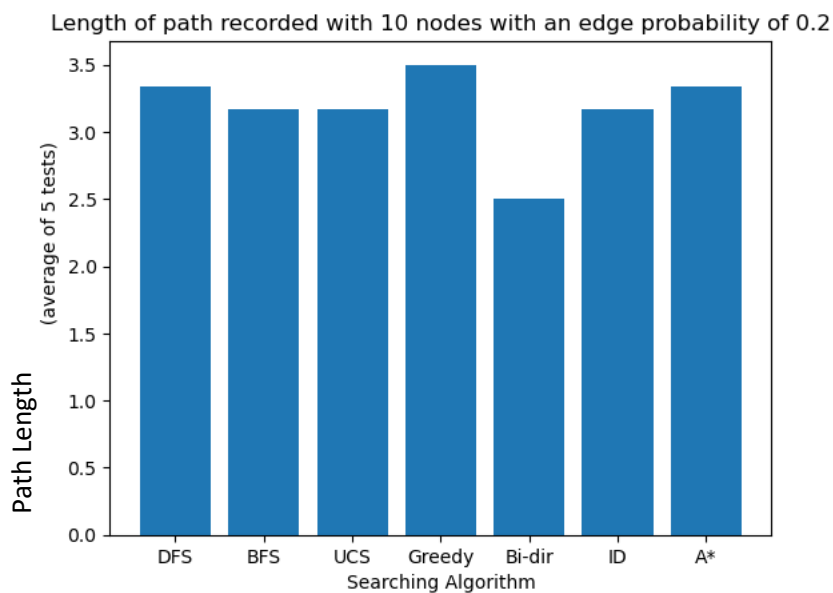
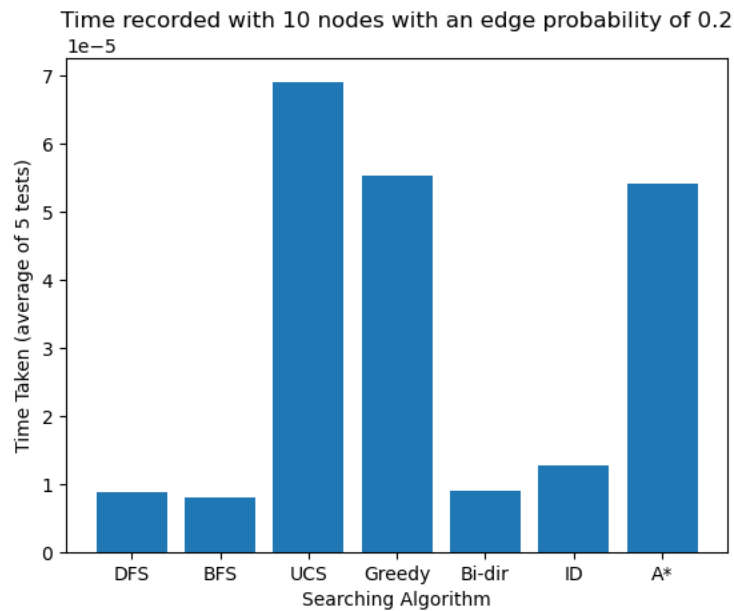
#### Introduction:

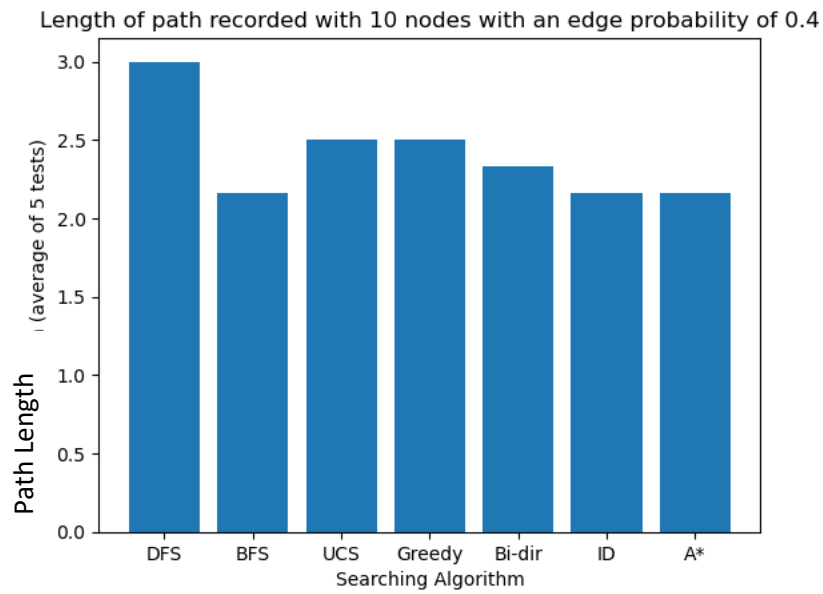
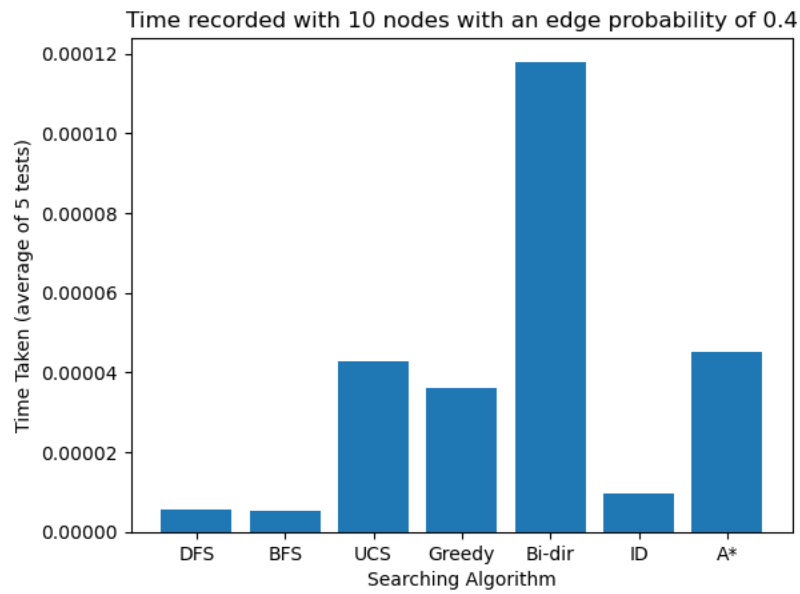
In this experiment, we explore the performance of various search algorithms in finding paths within randomly generated graphs. Each graph is created with a specified number of nodes (10, 20, 30, or 40) and varying edge probabilities (0.2, 0.4, 0.6, or 0.8), resulting in a total of 16 distinct graph configurations.

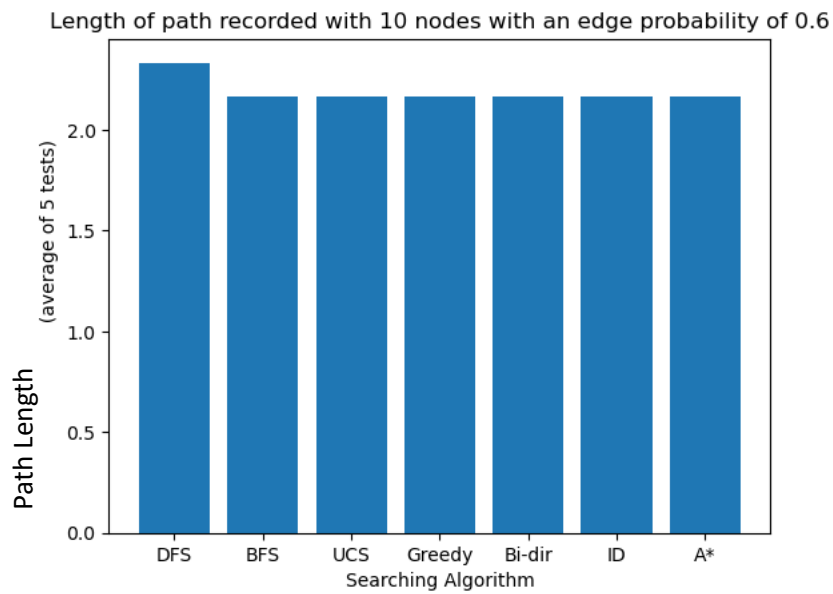
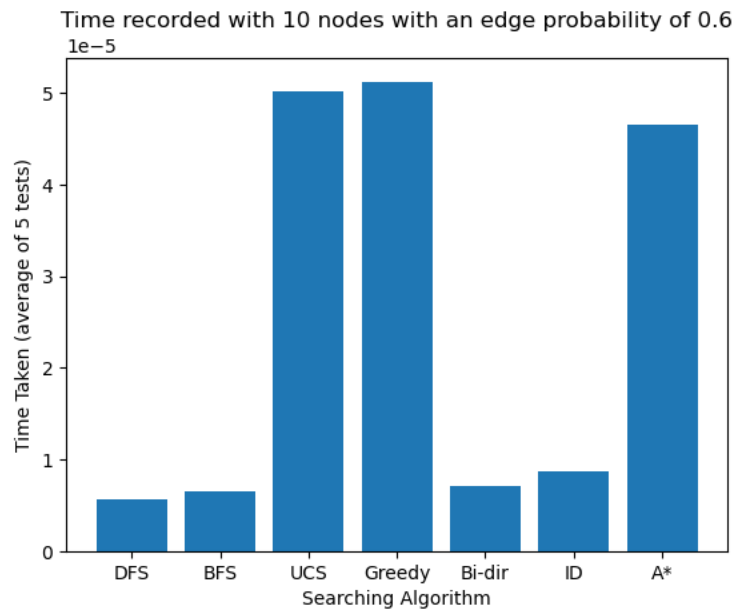
For each graph configuration, we conduct a series of tests where we randomly select 5 pairs of nodes. Then, utilizing each of the selected search algorithms - including DFS, BFS, UCS, Greedy

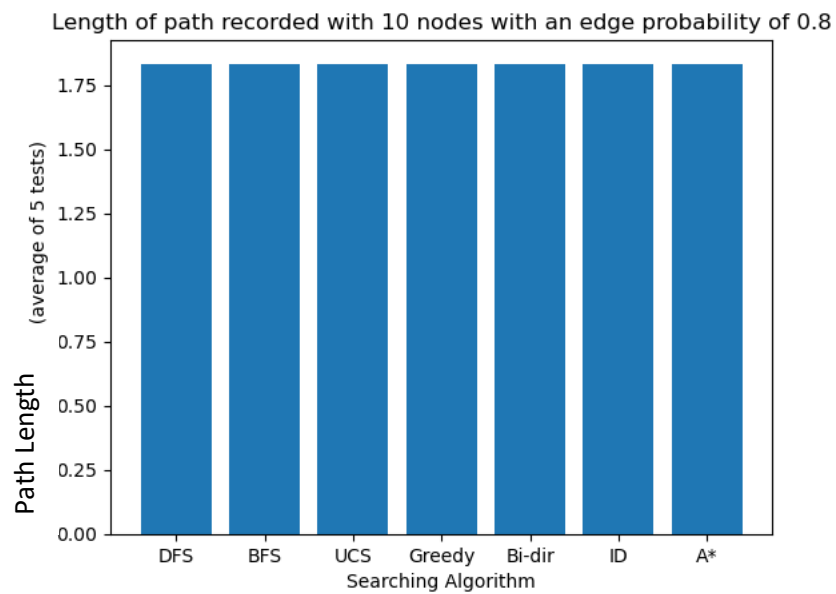
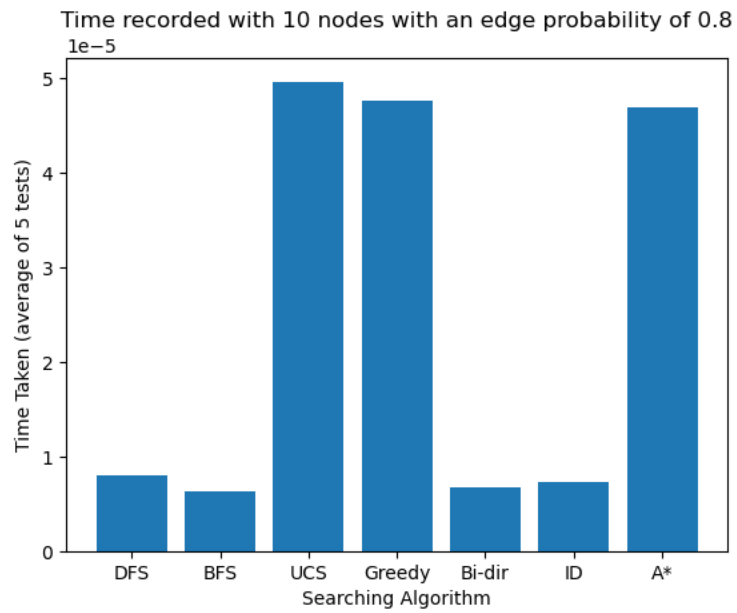
Search, Bidirectional Search, Iterative Deepening, and A\* Search - we measure the time taken to find paths between the selected node pairs. Additionally, we record the lengths of the paths found by each algorithm.

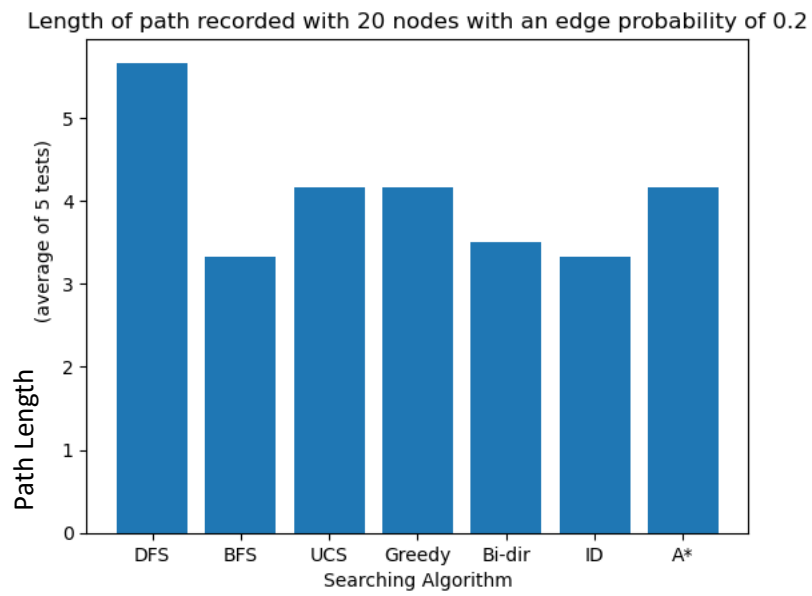
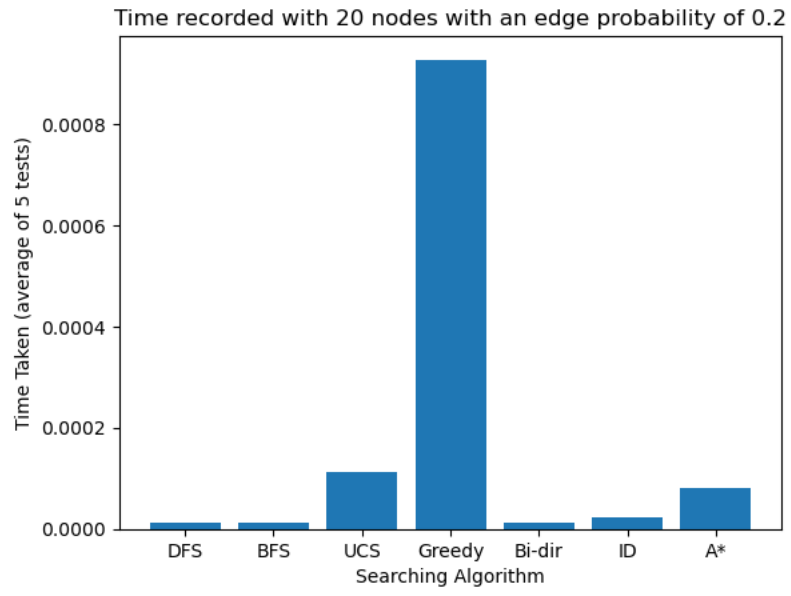
Through this experiment, we aim to investigate how the size of the graph and the density of edges influence the efficiency of the search algorithms. By examining the time taken and path lengths across different graph configurations, we seek to gain insights into the scalability and performance characteristics of these algorithms in varying search spaces.



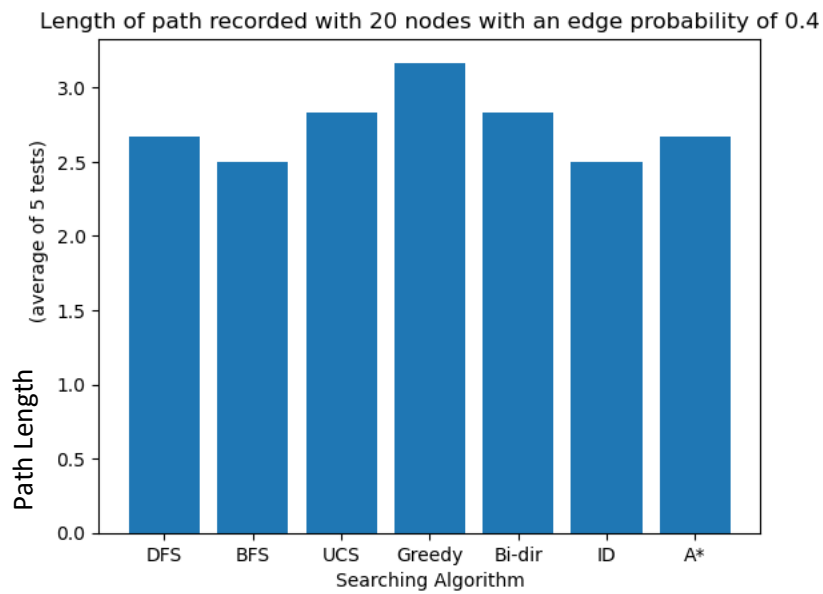
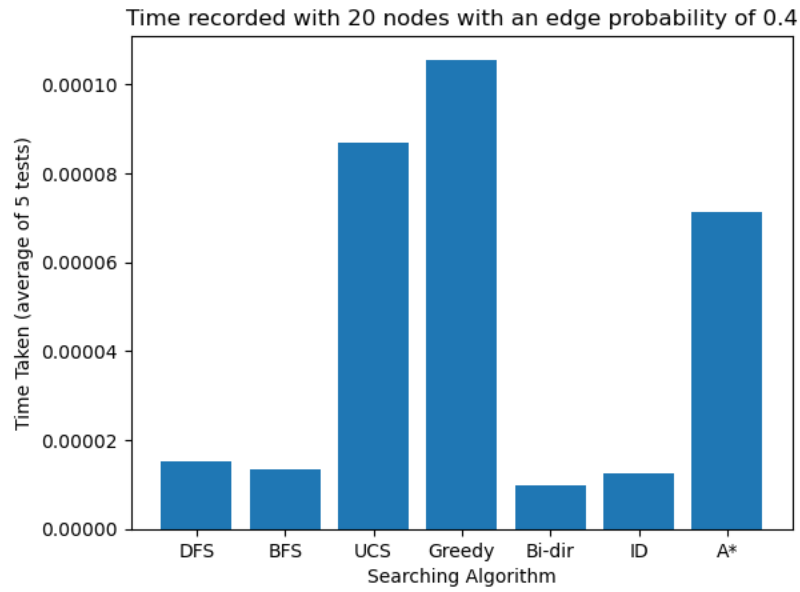


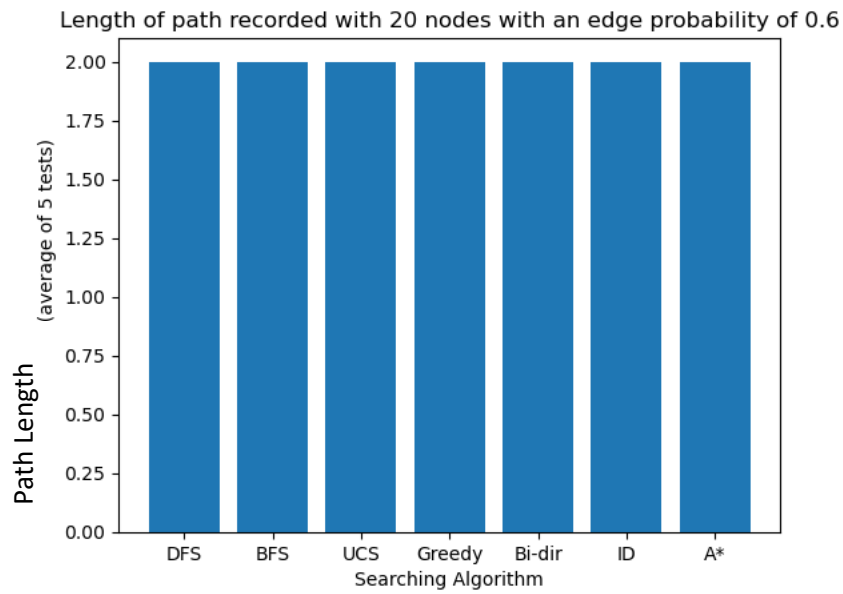
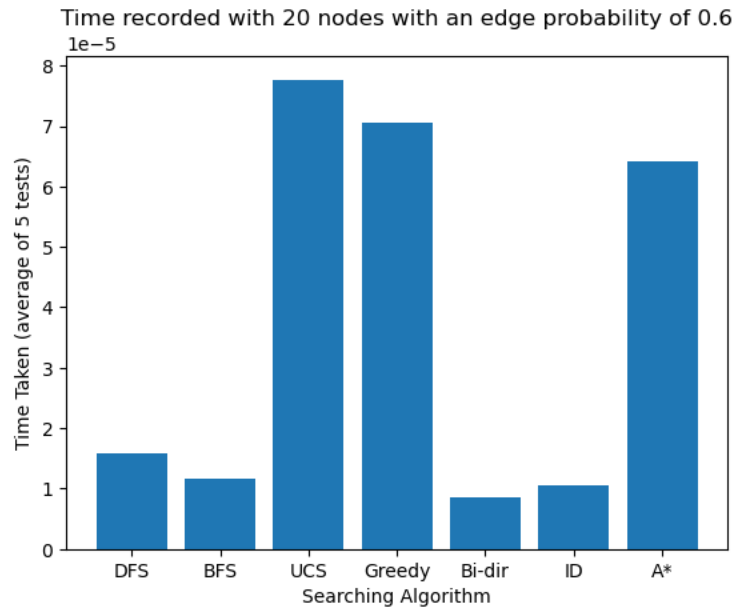


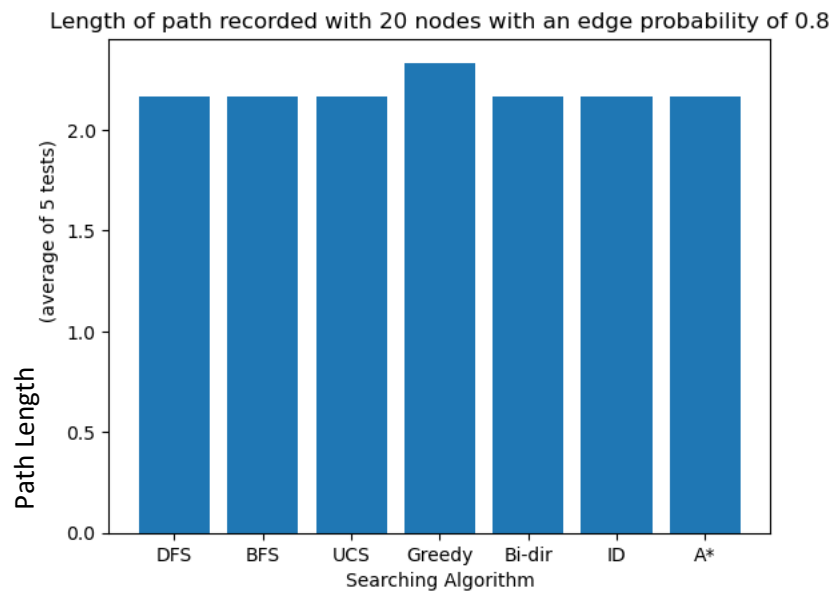
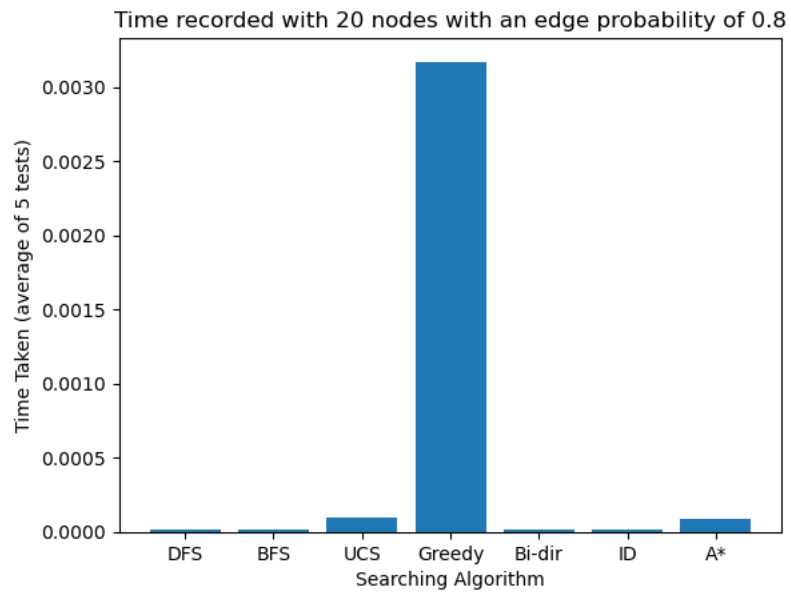


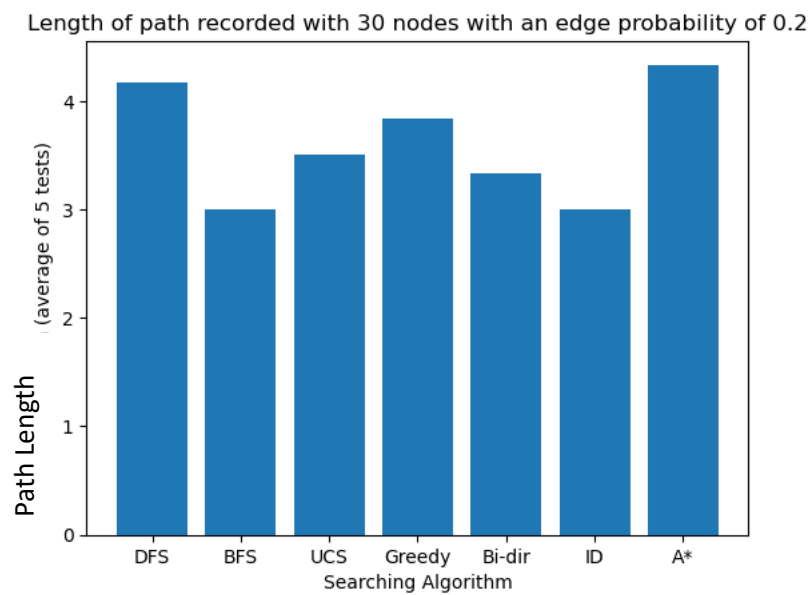
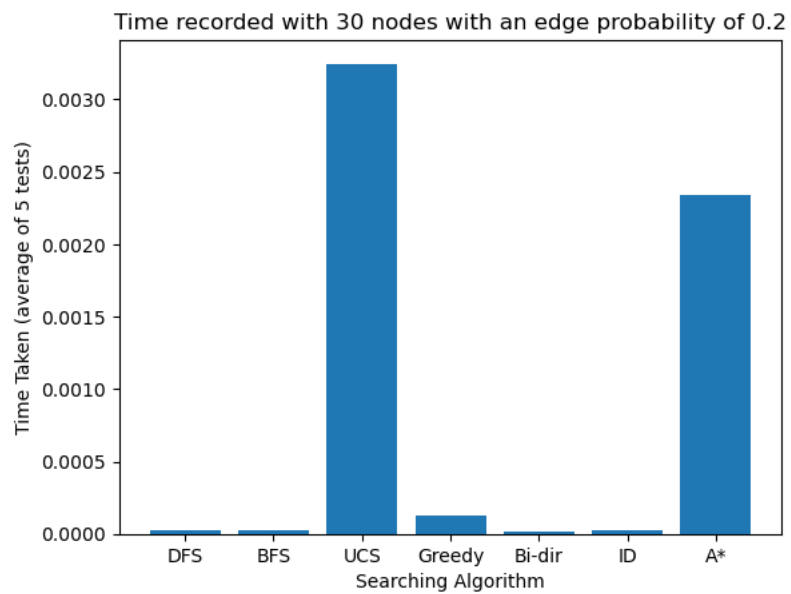


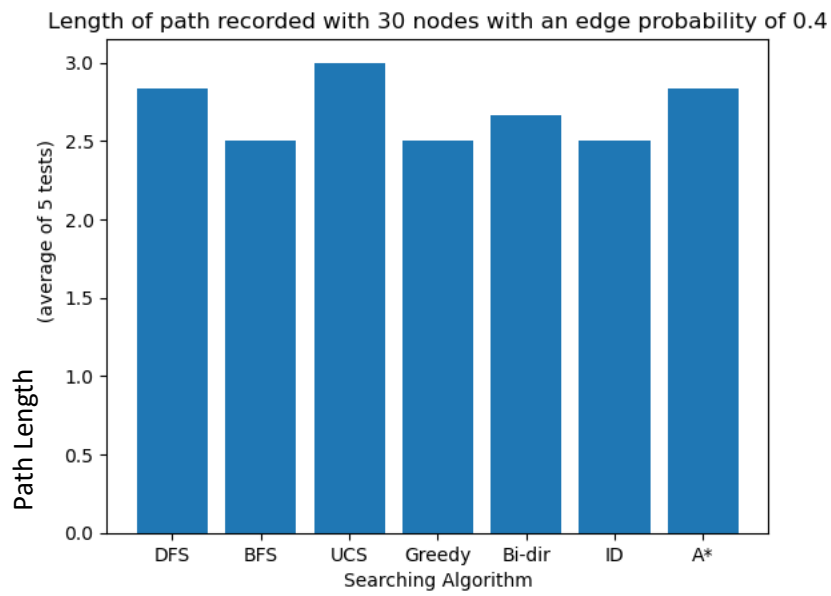
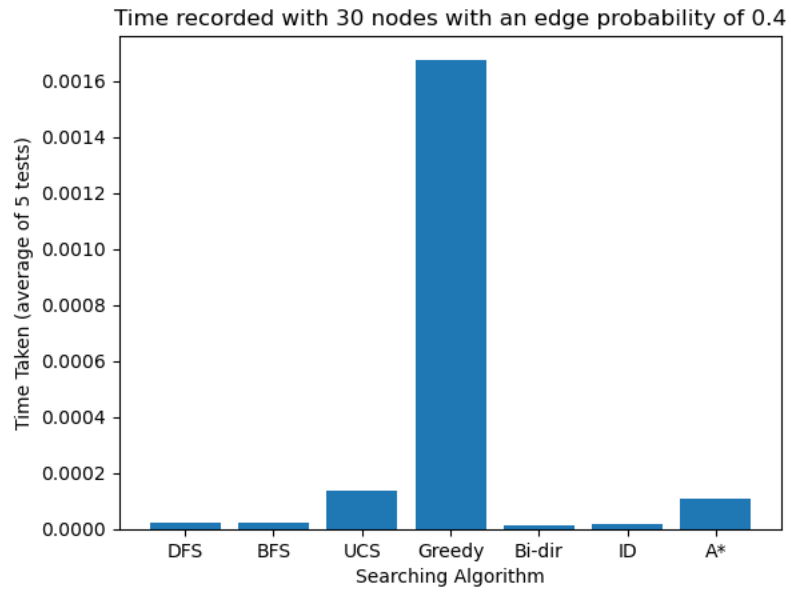


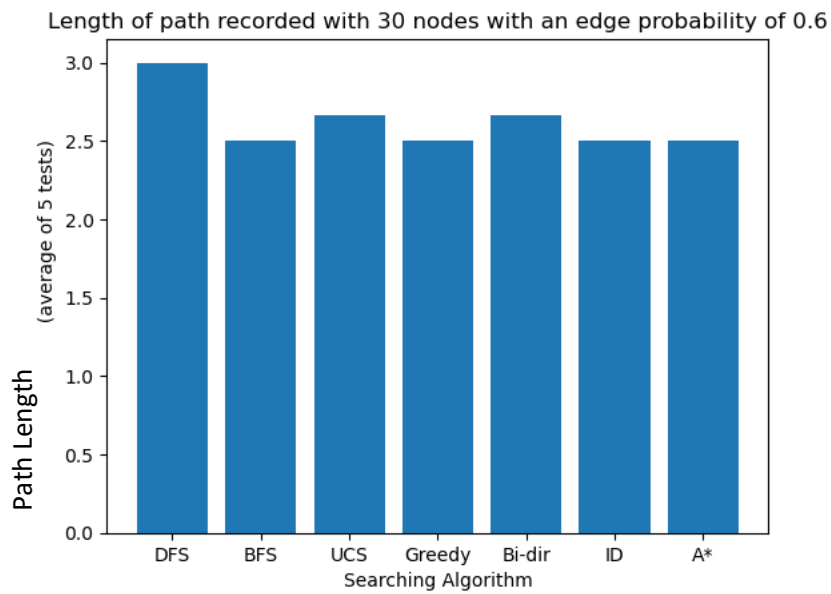
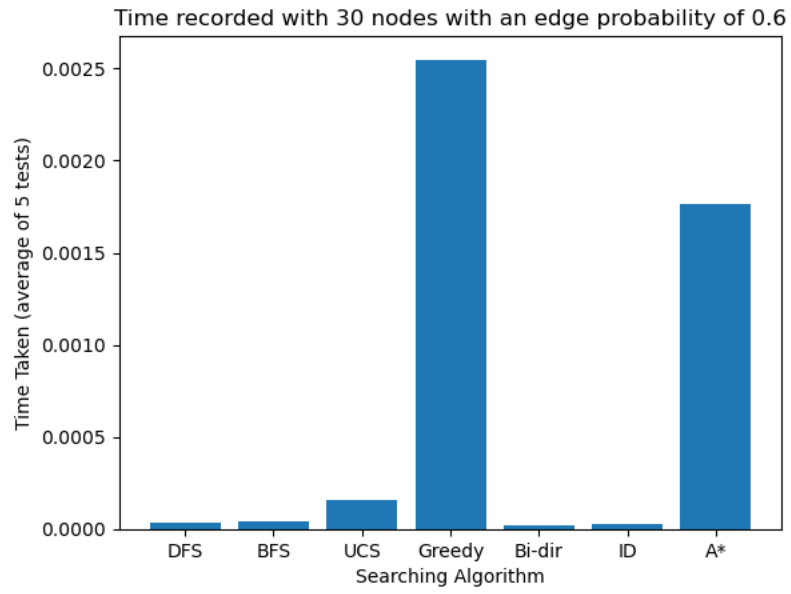


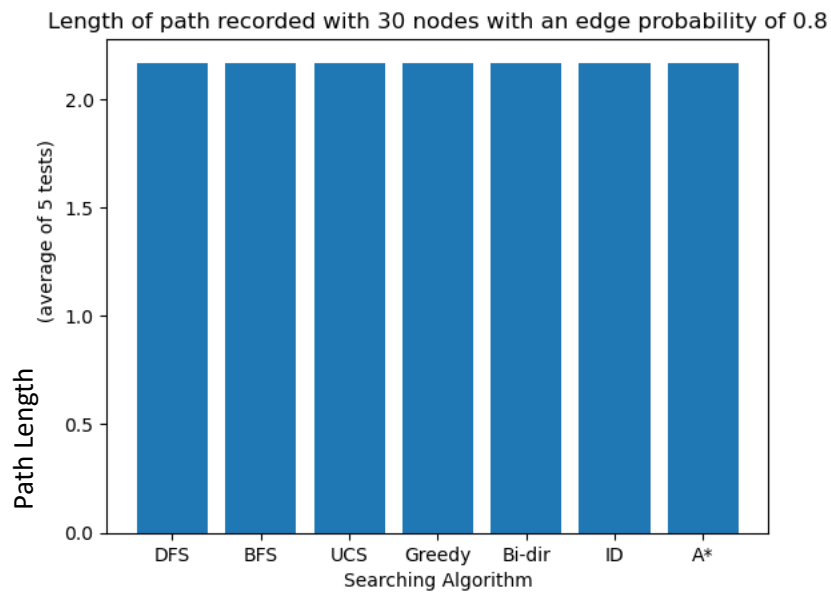
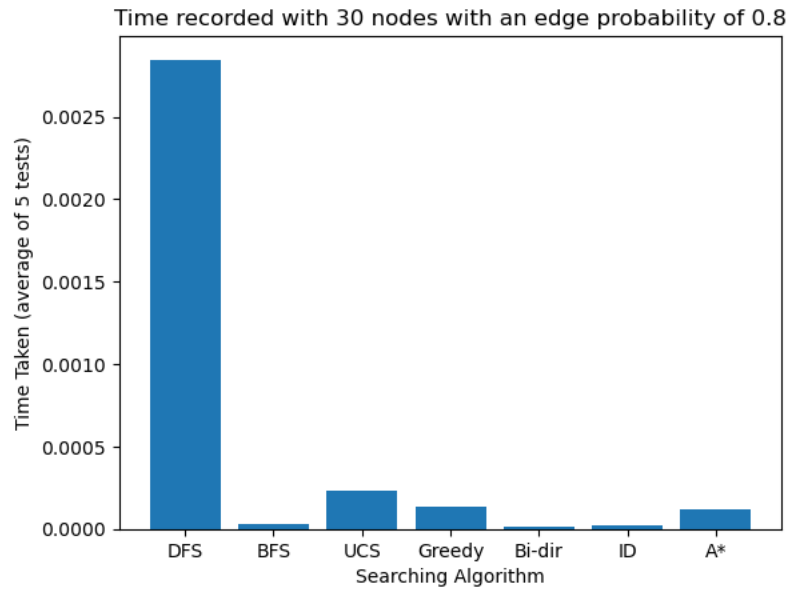


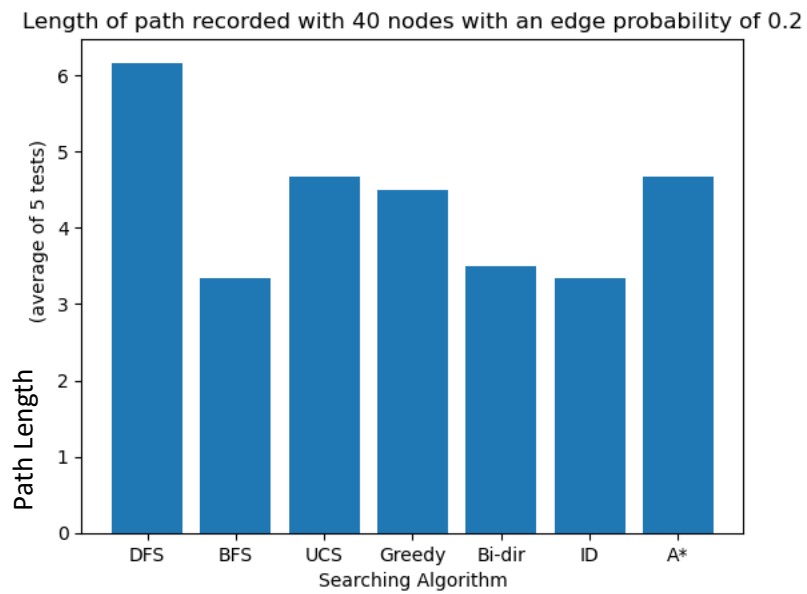
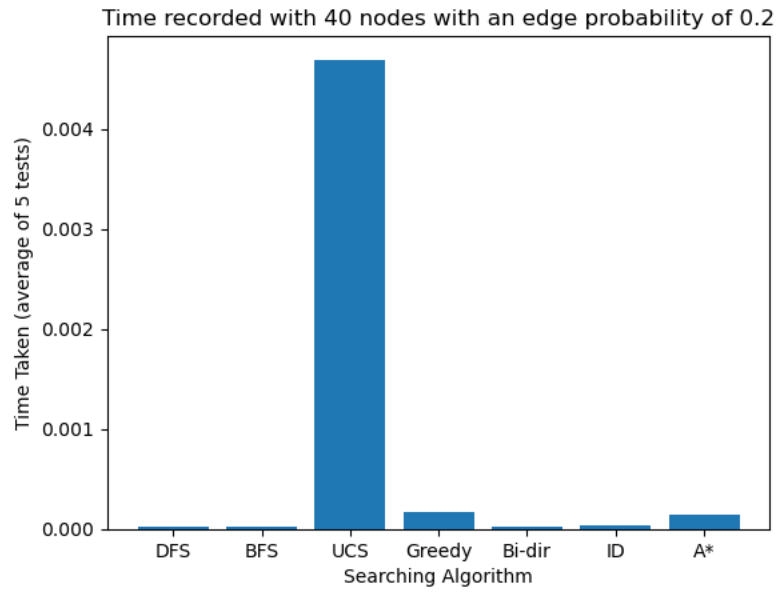




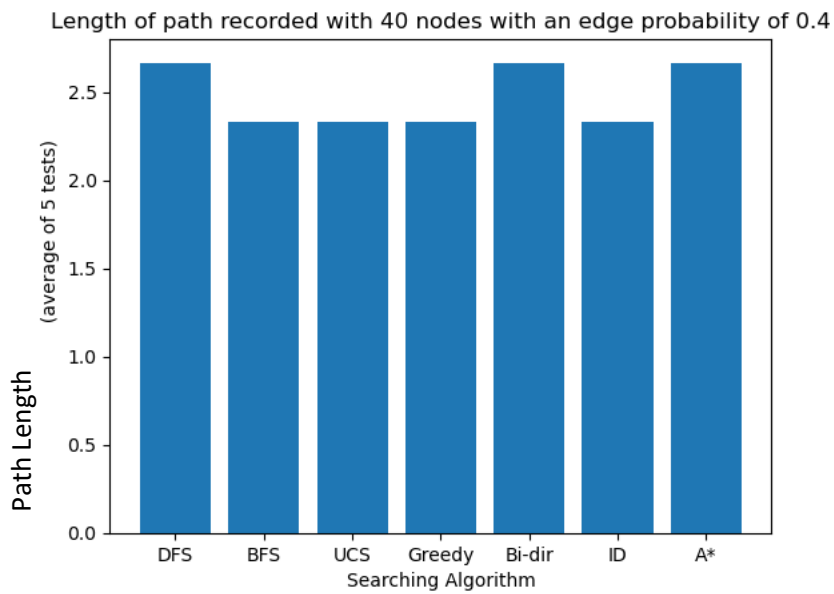
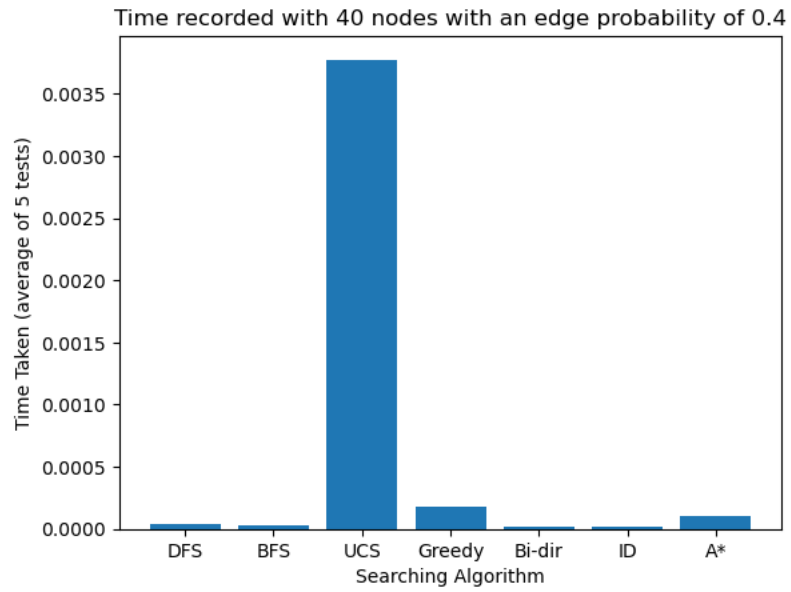


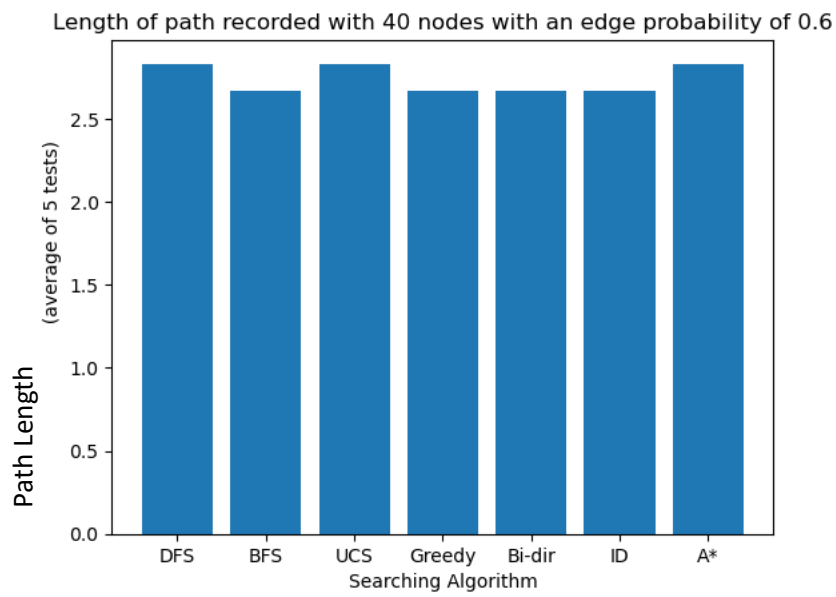
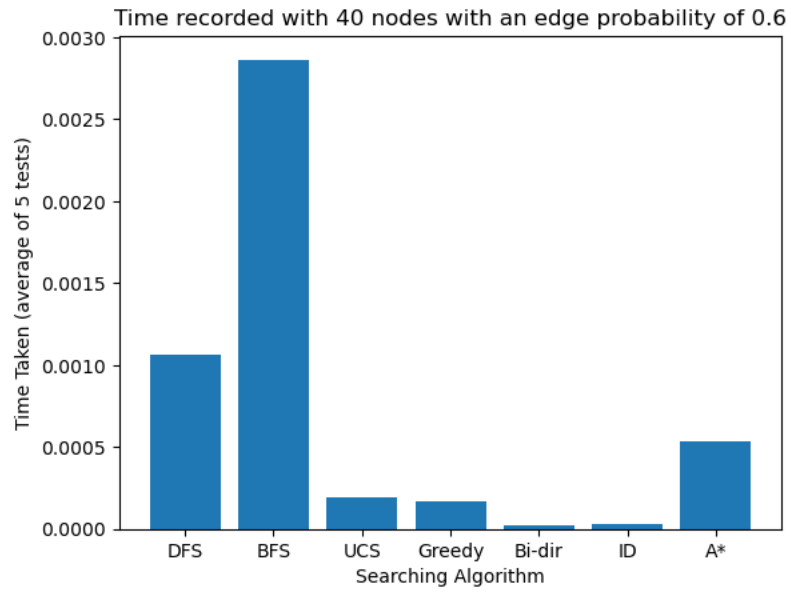


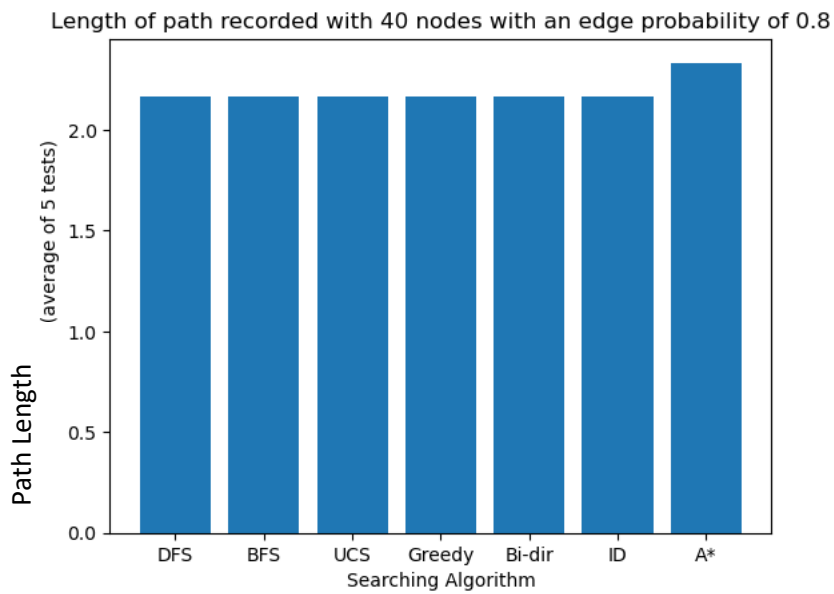
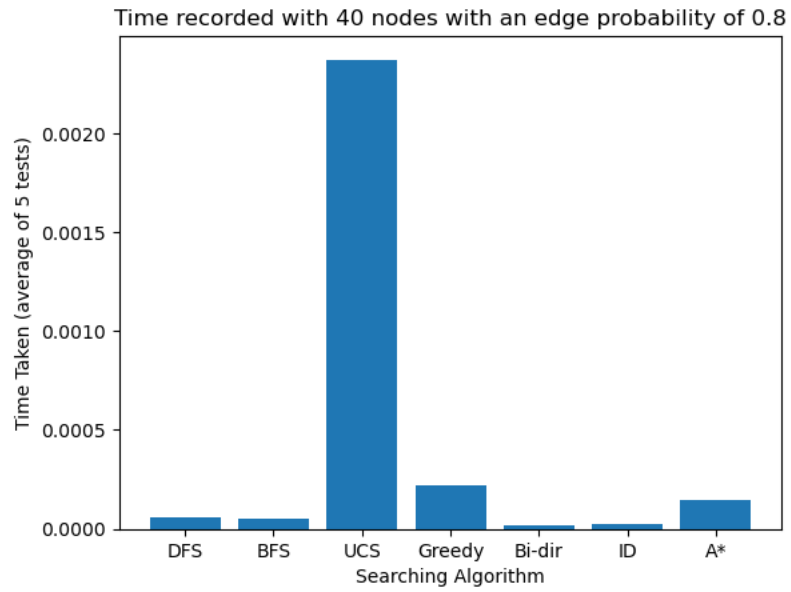




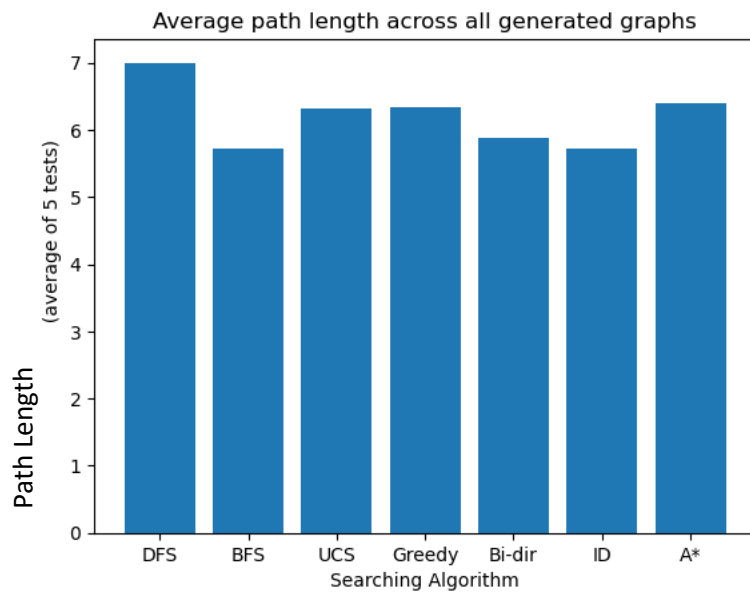
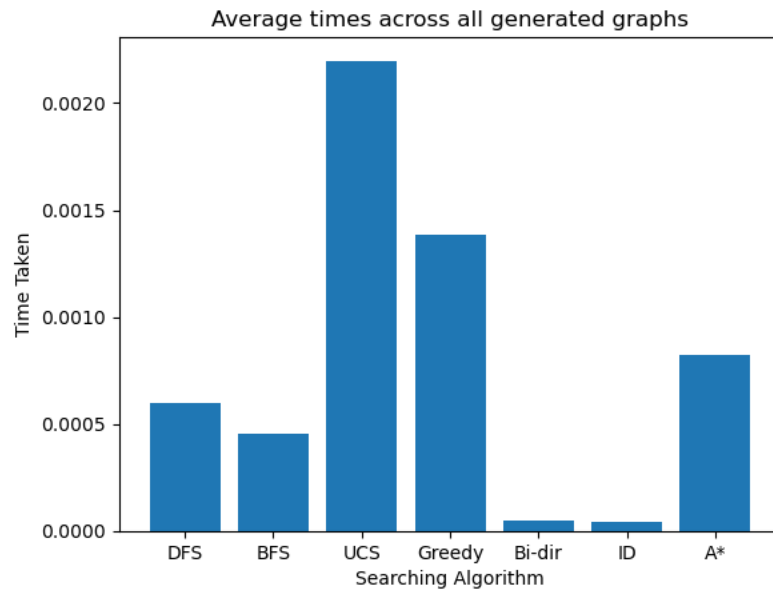




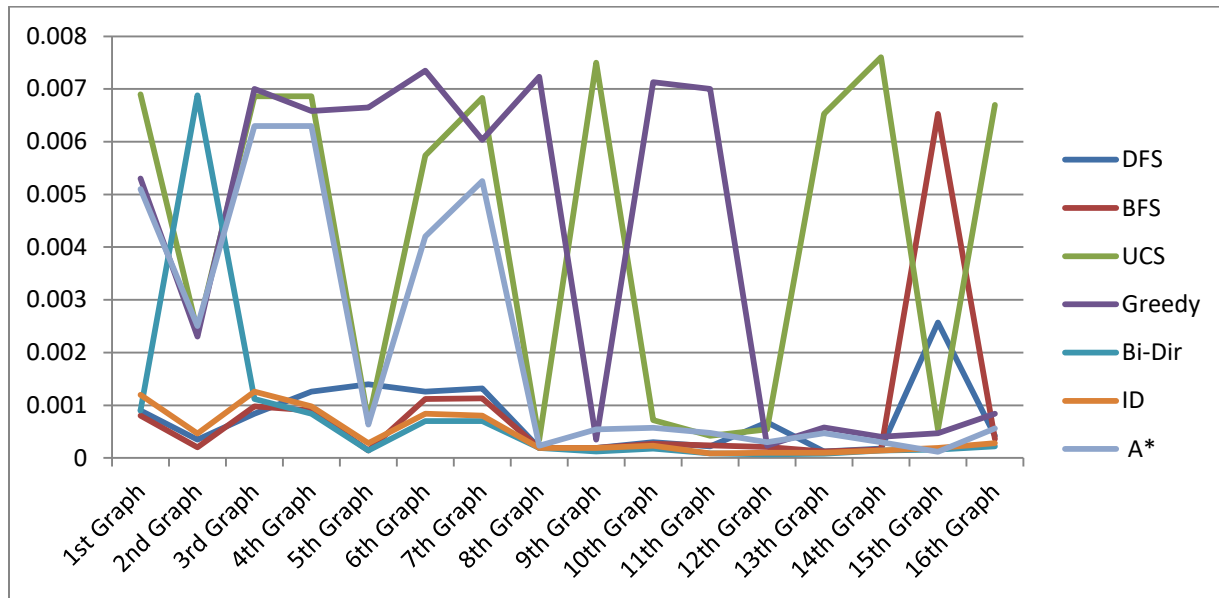




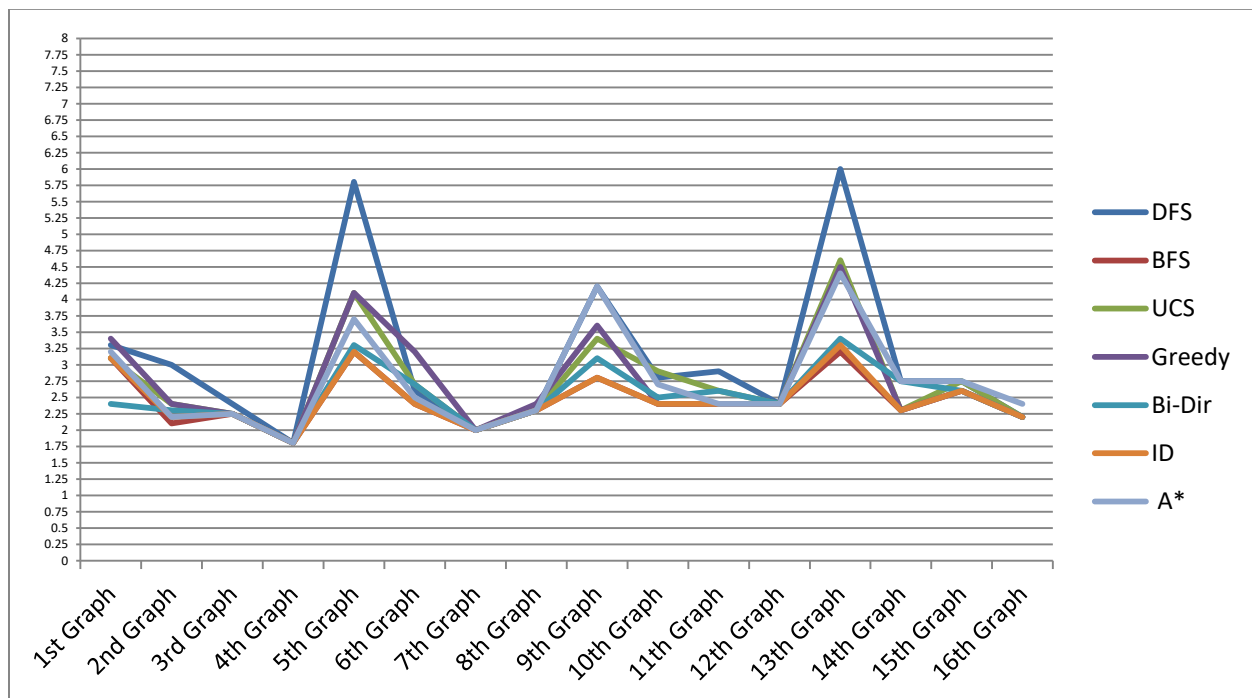
## Average Time and Path length across all graphs



**Average Time for various search algorithms across different graphs**



**Average path for various search algorithms across different graphs**



## Observations:

### 1. Effect of Edge Probability on Time and Path Length:

- As the edge probability increases, the time taken for most algorithms tends to increase, indicating that the algorithms take longer to search for paths in denser graphs.
- Conversely, the length of the paths tends to decrease as the edge probability increases, which makes sense as denser graphs offer more direct routes between nodes.

### 2. Performance of Algorithms:

- **Uninformed vs. Informed:** Uninformed search algorithms like DFS and BFS generally perform better in terms of path length compared to informed algorithms like A\* and greedy search, especially in denser graphs. However, informed algorithms often exhibit faster execution times, particularly in sparser graphs.
- **Greedy Search:** Greedy search consistently shows high path lengths across different graph densities, indicating that it tends to find suboptimal paths. However, it often performs well in terms of execution time, especially in sparser graphs.
- **UCS and A\*:** Uniform Cost Search (UCS) and A\* consistently exhibit competitive performance in both time and path length across different graph densities, showcasing their effectiveness in finding optimal paths efficiently.
- **Bidirectional Search:** Bidirectional search generally performs well in terms of time but may exhibit higher path lengths, especially in denser graphs.
- **Iterative Deepening (ID):** ID shows mixed performance, sometimes being comparable to informed algorithms like A\* and sometimes falling behind.

### 3. Graph Density Impact:

- In sparser graphs (lower edge probabilities), informed algorithms like A\* and UCS tend to have better performance in terms of time, while in denser graphs, their performance becomes comparable to or even worse than some uninformed algorithms.

- Uninformed algorithms like DFS and BFS tend to perform relatively better in terms of path length in denser graphs, possibly due to their systematic exploration of the graph structure.

#### 4. Trade-offs:

- There seems to be a trade-off between time and path length for different algorithms. Greedy algorithms prioritize speed over optimality, leading to shorter execution times but potentially longer paths. Informed algorithms, on the other hand, focus on finding optimal paths, which may require more time, especially in denser graphs.

## Question 3:

### Experiment 3: Measures Of Centrality

#### Introduction:

In this experiment, we analyze various measures of centrality for the given graph, including Degree, Closeness, Betweenness, Katz, Eigen-Vector centrality and Page-Rank Centrality. Each centrality measure provides insights into the importance or influence of nodes within the graph based on different criteria, such as their connectivity, proximity, or role in facilitating communication. By examining these centrality measures, we gain a deeper understanding of the structural properties and dynamics of the network under study.

The table provided contains the rankings of each city under different centrality measures.

	Degree Centrality	Closeness Centrality	Betweenness Centrality	Katz Centrality	Eigen-Vector Centrality	Page-Rank Centrality
1	Sibiu	Pitesti	Bucharest	Sibiu	Sibiu	Bucharest
2	Bucharest	Rimnicu_Vilcea	Urziceni	Oradea	Arad	Sibiu
3	Arad	Bucharest	Sibiu	Drobeta	Oradea	Urziceni
4	Craiova	Sibiu	Pitesti	Urziceni	Fagaras	Arad
5	Rimnicu_Vilcea	Craiova	Craiova	Vaslui	Rimnicu_Vilcea	Craiova
6	Pitesti	Fagaras	Fagaras	Zerind	Zerind	Pitesti
7	Urziceni	Urziceni	Vaslui	Neamt	Bucharest	Rimnicu_Vilcea
8	Oradea	Giurgiu	Drobeta	Giurgiu	Pitesti	asi

9	Zerind	Drobeta	Mehadia	Pitesti	Craiova	Hirsova
10	Timisoara	Mehadia	Arad	Hirsova	Timisoara	Vaslui
11	Lugoj	Lugoj	Hirsova	Rimnicu_Vilcea	Urziceni	Lugoj
12	Mehadia	Arad	Iasi	Arad	Drobeta	Mehadia
13	Drobeta	Hirsova	Lugoj	Bucharest	Vaslui	Timisoara
14	Fagaras	Oradea	Rimnicu_Vilcea	Fagaras	Giurgiu	Drobeta
15	Hirsova	Timisoara	Timisoara	Eforie	Lugoj	Zerind
16	Vaslui	Zerind	Oradea	Mehadia	Hirsova	Oradea
17	Iasi	Vaslui	Zerind	Lugoj	Mehadia	Fagaras
18	Giurgiu	Eforie	Giurgiu	Iasi	Iasi	Neamt
19	Eforie	Iasi	Eforie	Craiova	Eforie	Eforie
20	Neamt	Neamt	Neamt	Timisoara	Neamt	Giurgiu

## Observations:

### ❖ Consistent Top-Ranking Cities:

- Cities like Bucharest, Sibiu, and Pitesti consistently rank high across multiple centrality measures.
- This suggests their pivotal roles in facilitating communication, information flow, and overall connectivity within the network.

### ❖ Peripheral or Less Influential Nodes:

- Conversely, cities with lower centrality rankings may represent peripheral or less influential nodes within the network.
- This may indicate their relative isolation or limited interactions with other cities.
- Some cities with lower centrality rankings include: Giurgiu, Eforie, Neamt, Iasi, Timisoara, Lugoj, Mehadia, Drobeta and Vaslui.

### ❖ Diverse Characteristics Across Measures:

- Comparing rankings across different centrality measures reveals cities with diverse characteristics in their centrality profiles.
- For instance, Arad ranks significantly high in Degree Centrality (#3), indicating it has many direct connections, but it ranks lower in Betweenness Centrality (#10), suggesting it may not serve as a crucial bridge between other cities.
- This implies that while Arad is well-connected locally, it may not play a pivotal role in facilitating communication and information flow between other cities, potentially limiting its influence in the overall network dynamics.



## Question 4:

The following tables show the time and space complexities of various operations under multiple graph representations. Note that 'V' and 'E' represent the number of vertices and edges for a graph respectively.

### Time complexity for operations

Operations	Graph Representations		
	Adjacency Matrix	Adjacency List	Edge List
Insertion of vertices	$O(V^2)$	$O(1)$	-
Insertion of Edges	$O(1)$	$O(1)$	$O(1)$
Deletion of vertices	$O(V^2)$	$O( V  +  E )$	$O(E)$
Deletion of Edges	$O(1)$	$O( E )$	$O(E)$
Checking for the existence of an edge	$O(1)$	$O(E)$	$O(E)$
Finding the neighbors of a vertex	$O(V)$	$O(1)$	$O(E)$

### Space complexity for different graph representations:

	Graph Representations		
	Adjacency Matrix	Adjacency List	Edge List
Space complexities	$O(V^2)$	$O( V  +  E )$	$O(E)$