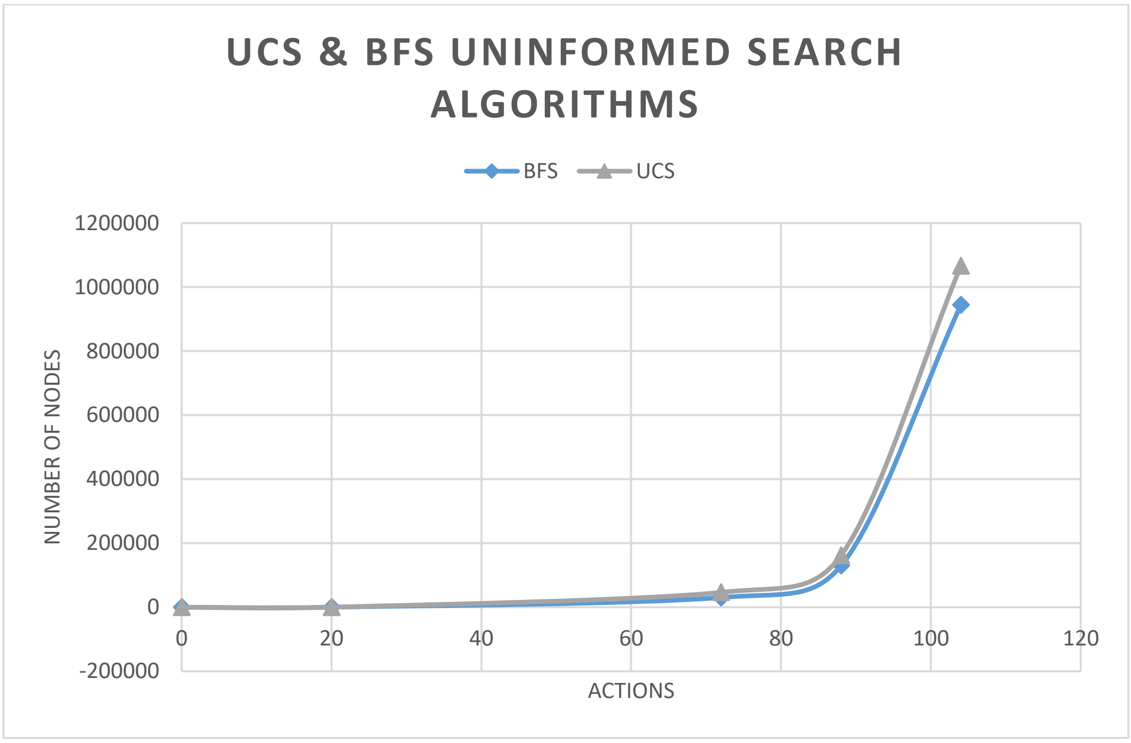
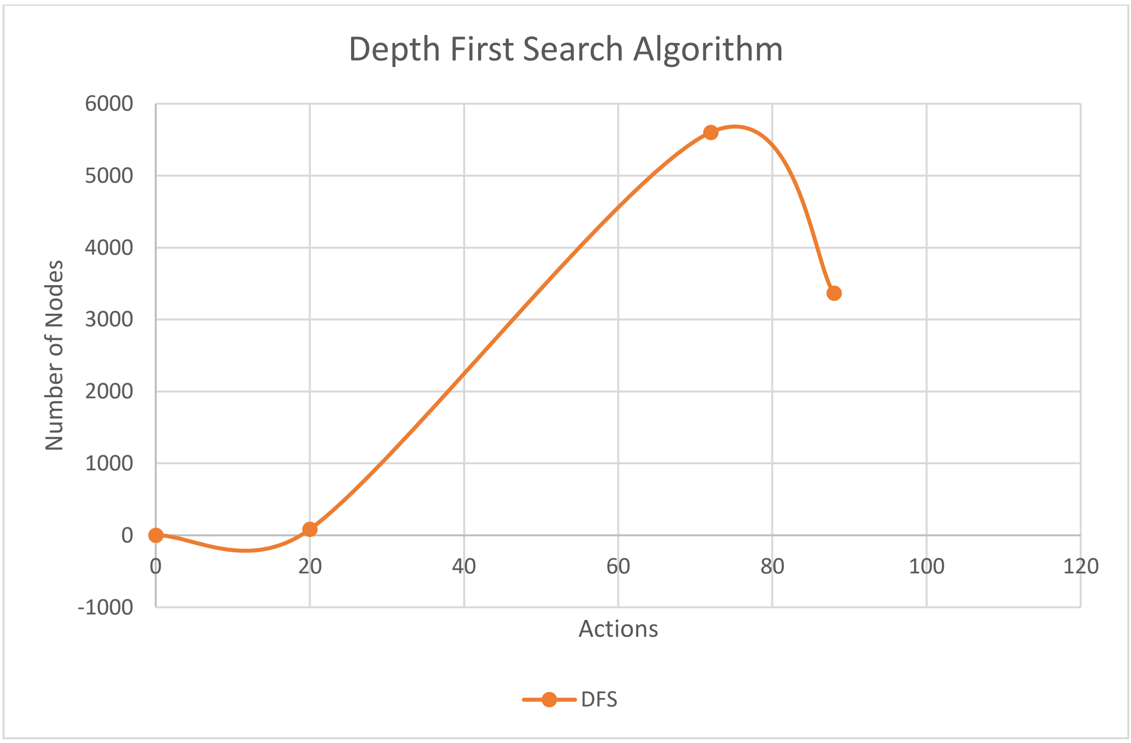
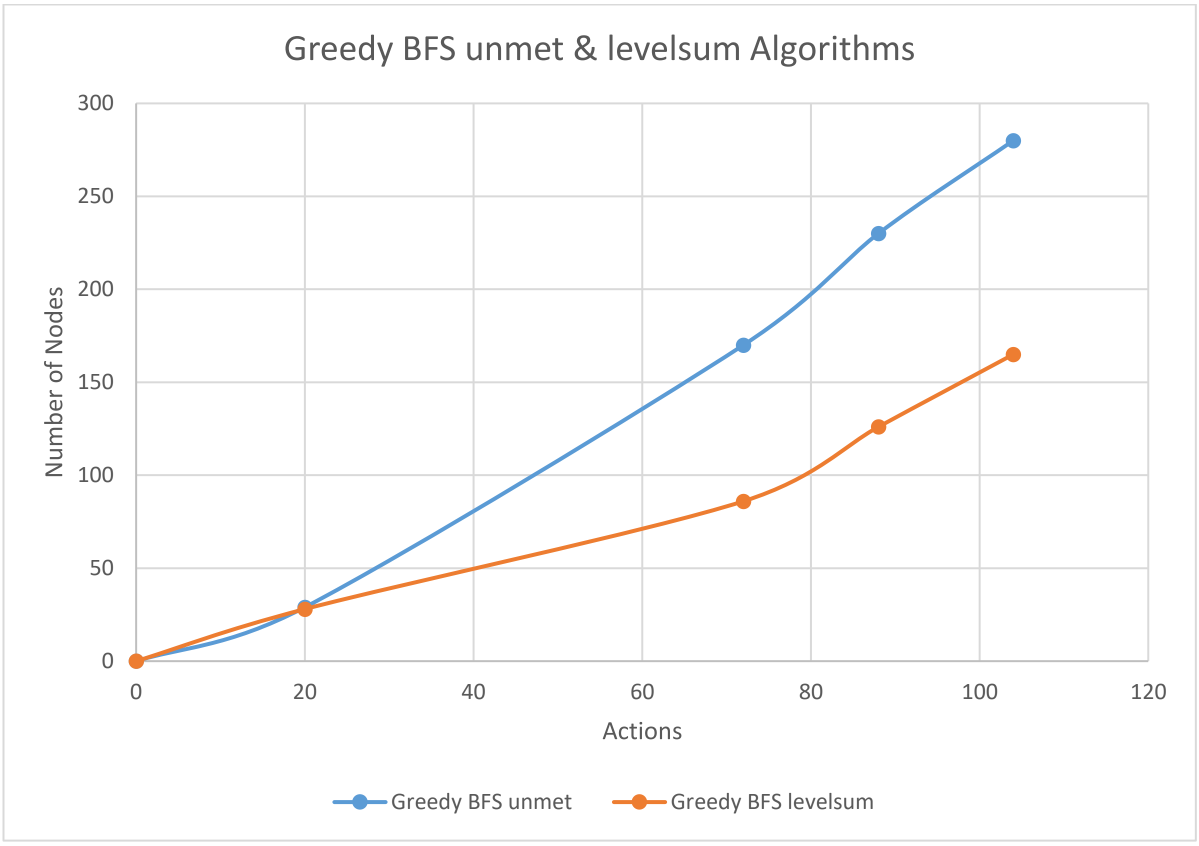
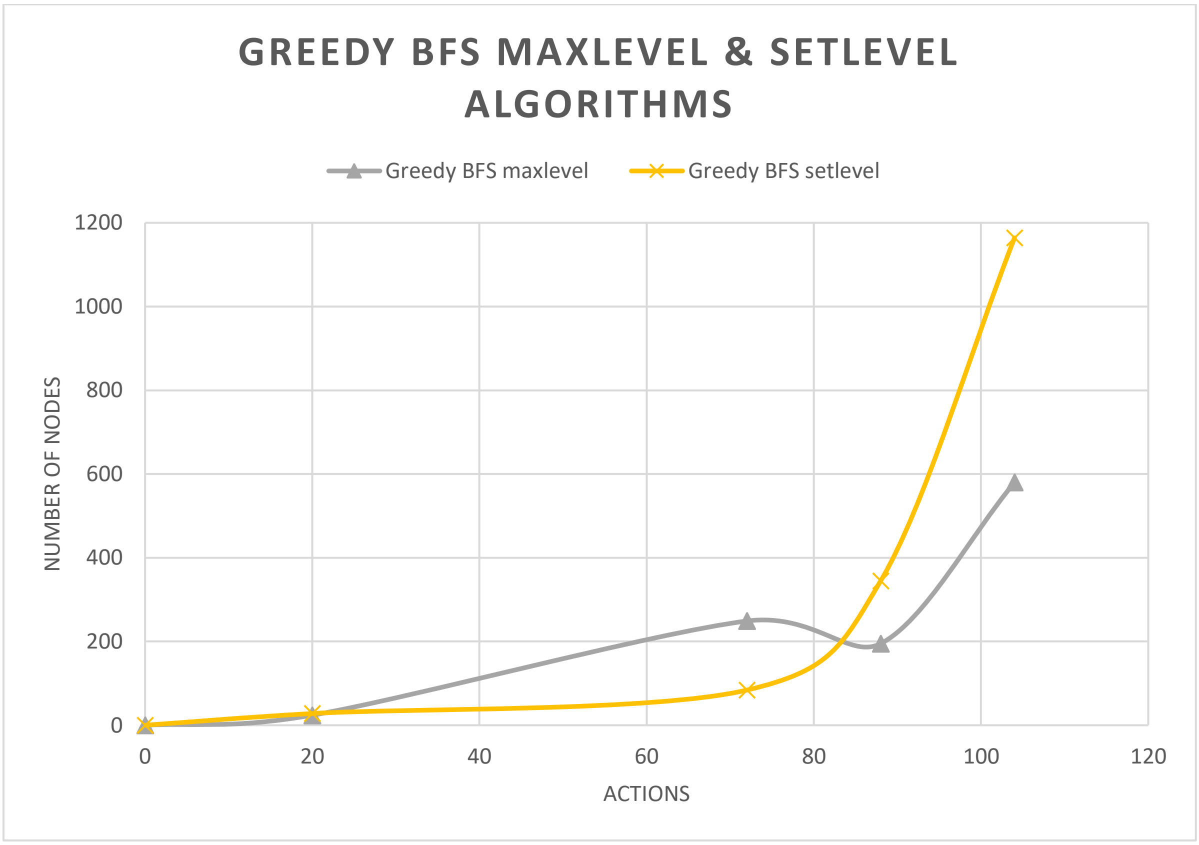
Number of nodes expanded against the number of actions in the domain



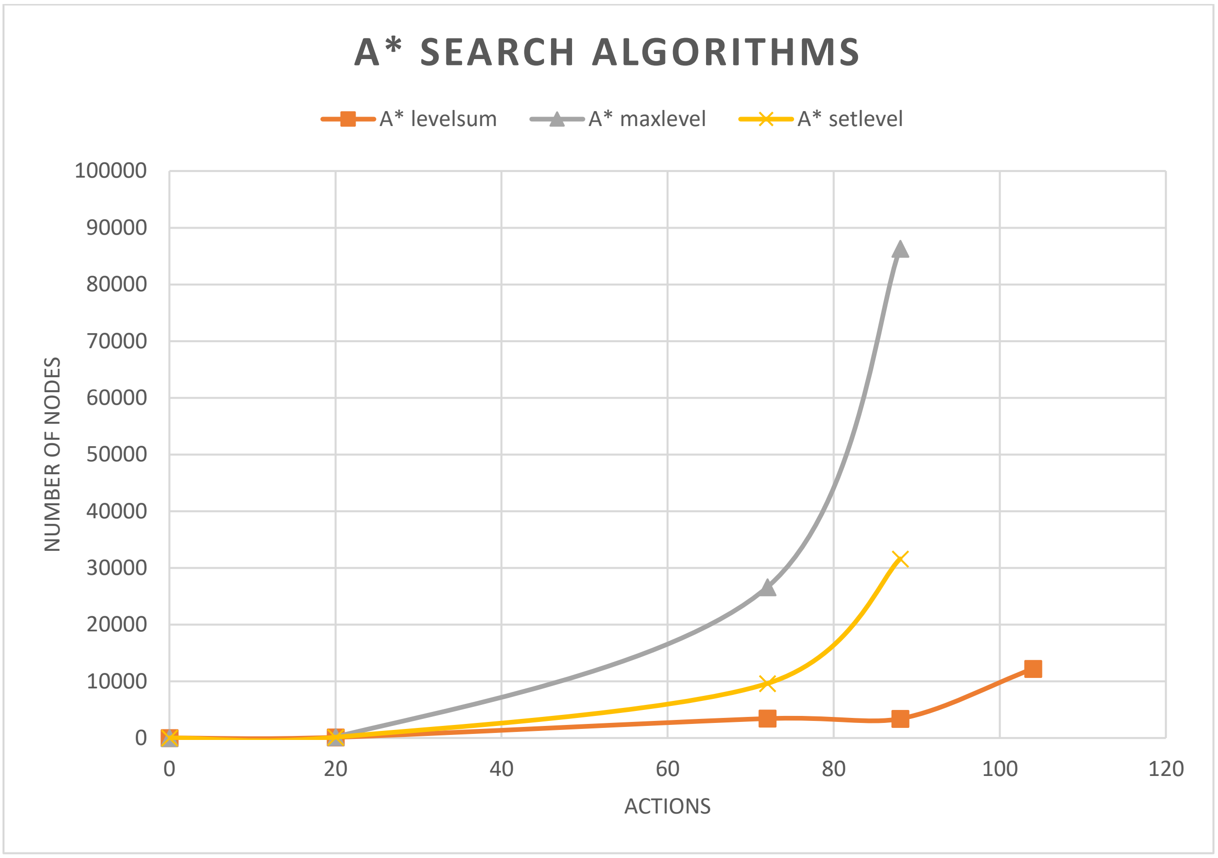


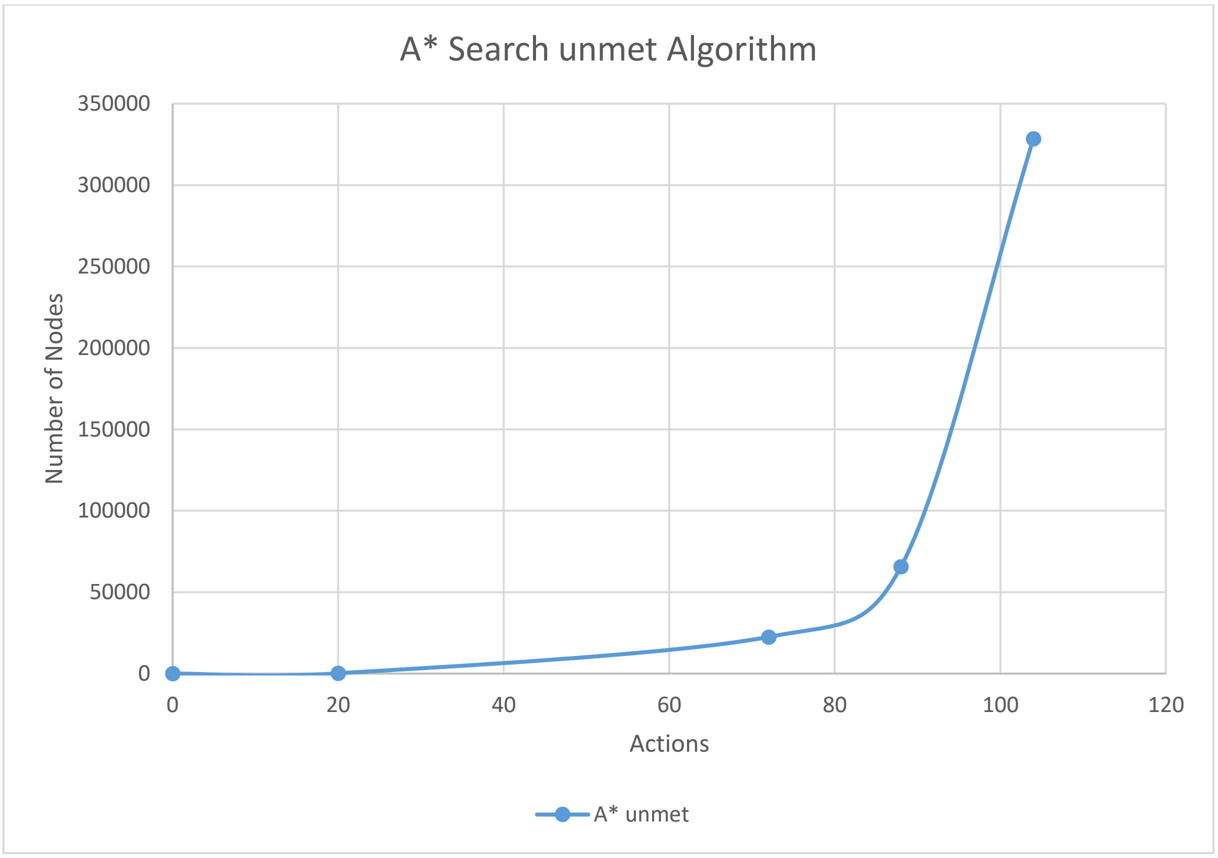


Greedy BFS unmet & levelsum Algorithms
Number of Nodes
300
250
200
150
100
50
0
0
20
40
60
80
100
120
Actions
Greedy BFS unmet
Greedy BFS levelsum



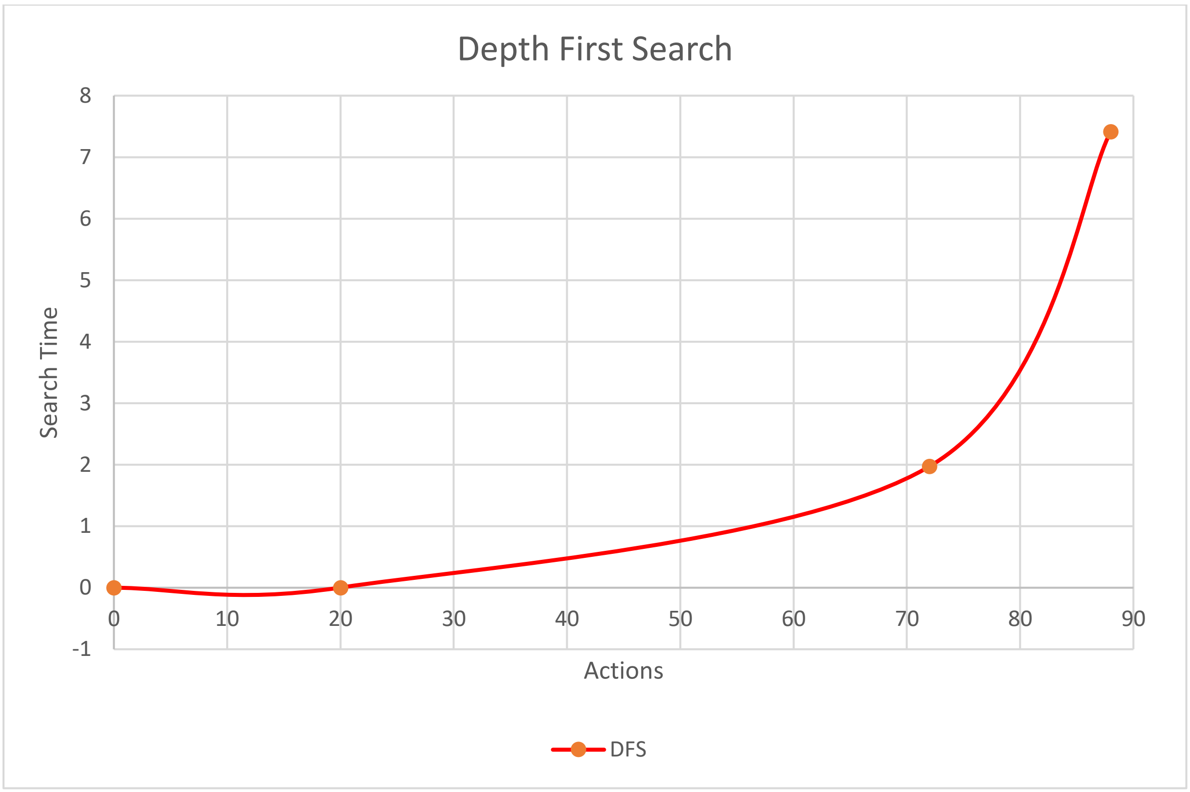
GREEDY BFS MAXLEVEL & SETLEVEL ALGORITHMS
Greedy BFS maxlevel
Greedy BFS setlevel
NUMBER OF NODES
1200
1000
800
600
400
200
0
0
20
40
60
80
100
120
ACTIONS

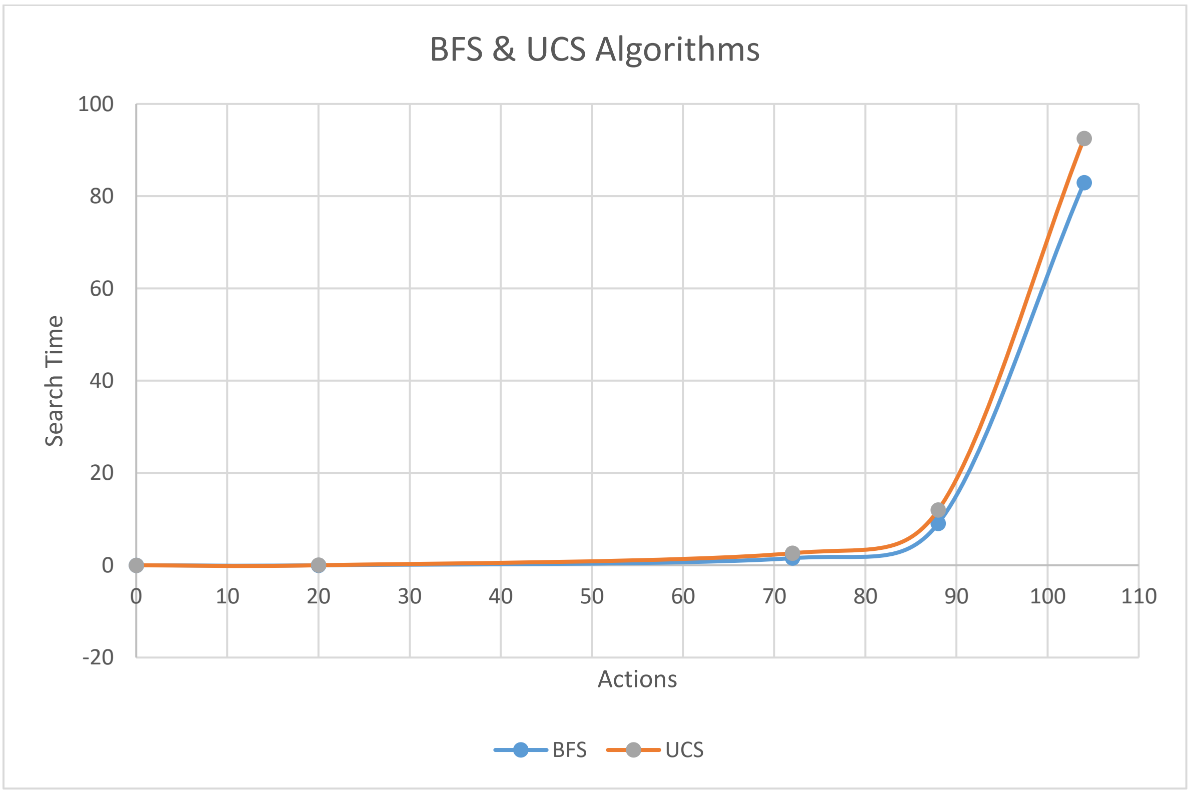

A* SEARCH ALGORITHMS
A* levelsum
A* maxlevel
A* setlevel
NUMBER OF NODES
100000
90000
80000
70000
60000
50000
40000
30000
20000
10000
0
0
20
40
60
80
100
120
ACTIONS

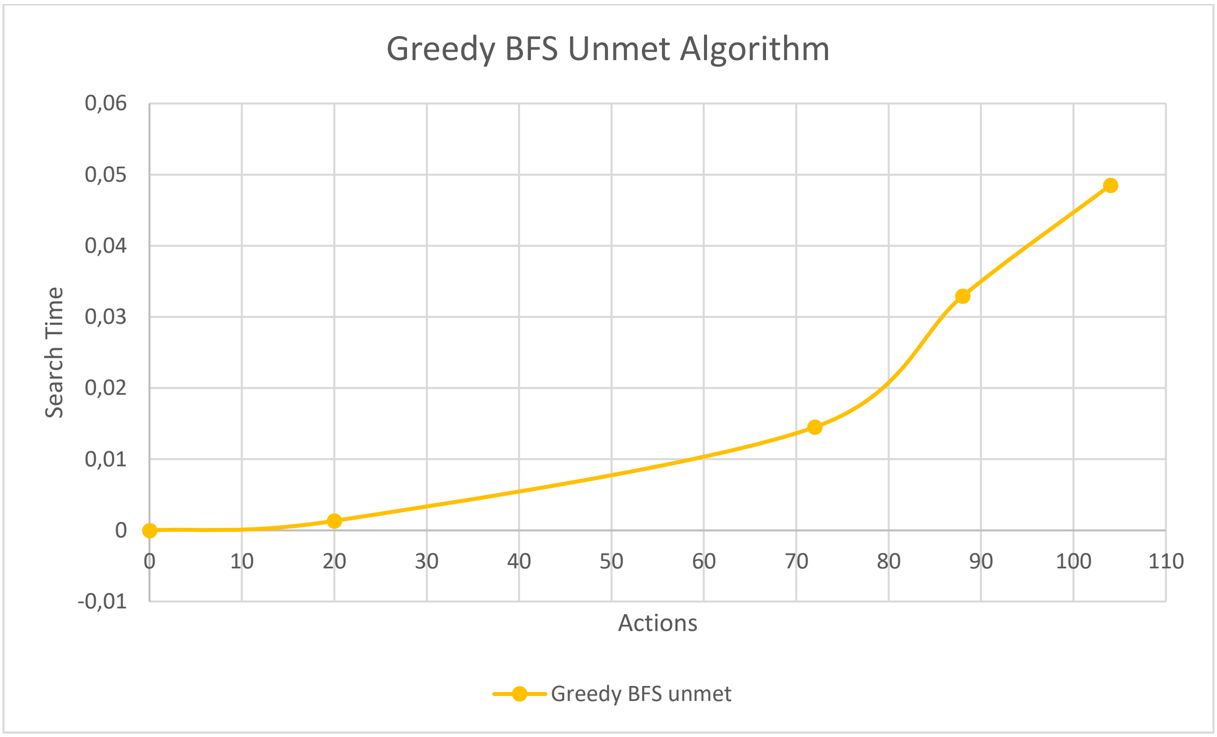

A* Search unmet Algorithm
Number of Nodes
350000
300000
250000
200000
150000
100000
50000
0
0
20
40
60
80
100
120
Actions
A* unmet

In the uninformed search algorithms, the number of nodes increased very fast as the number of as the problem size increased, when there were 20 actions BFS had expanded 178 nodes, UCS expanded 240 nodes and DFS expanded 84 nodes, however as the actions went up the number of nodes increased substantially from 240 to 46618 for UCS, which is nearly a 200 time increase.
For the Greedy best first search algorithms we see that the “levelsum” algorithm had the least number of nodes expanded and the “setlevel” algorithm had the most nodes expanded as the problem size increased. In A* search the “unmet” algorithm had the most nodes expanded with a value of 328,509 and “levelsum” had the least nodes expanded as the problem size increased.

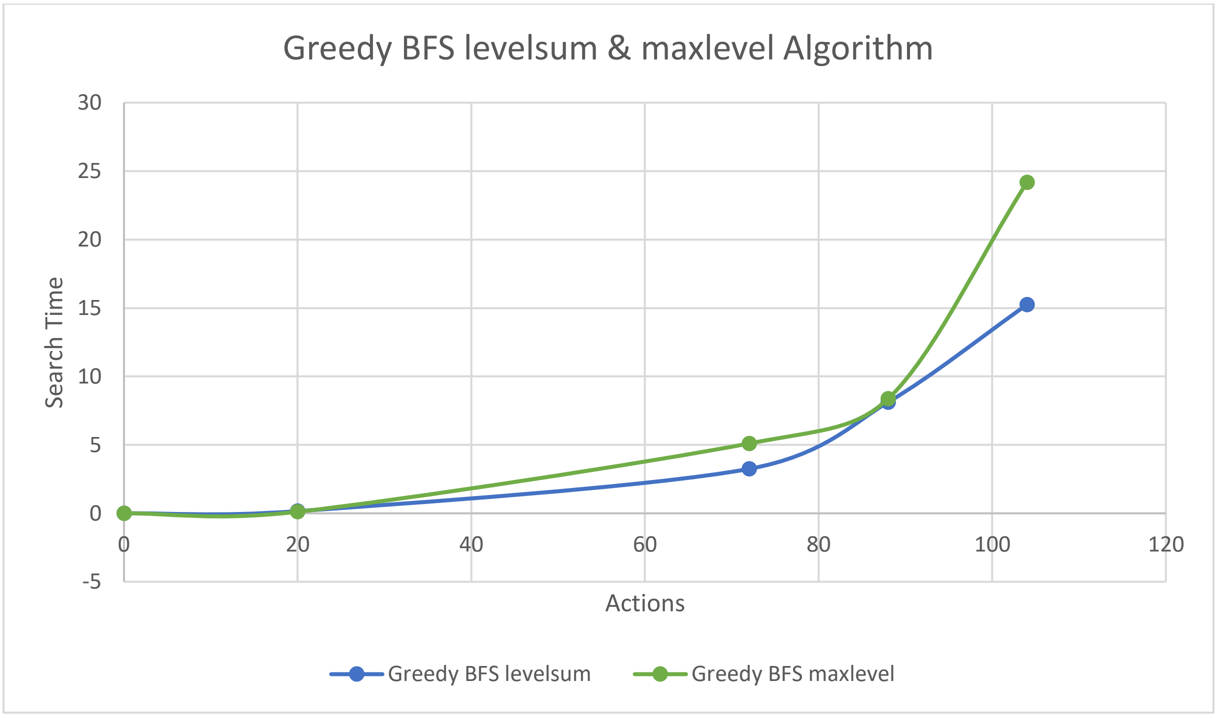
Search time against the number of actions in the domain



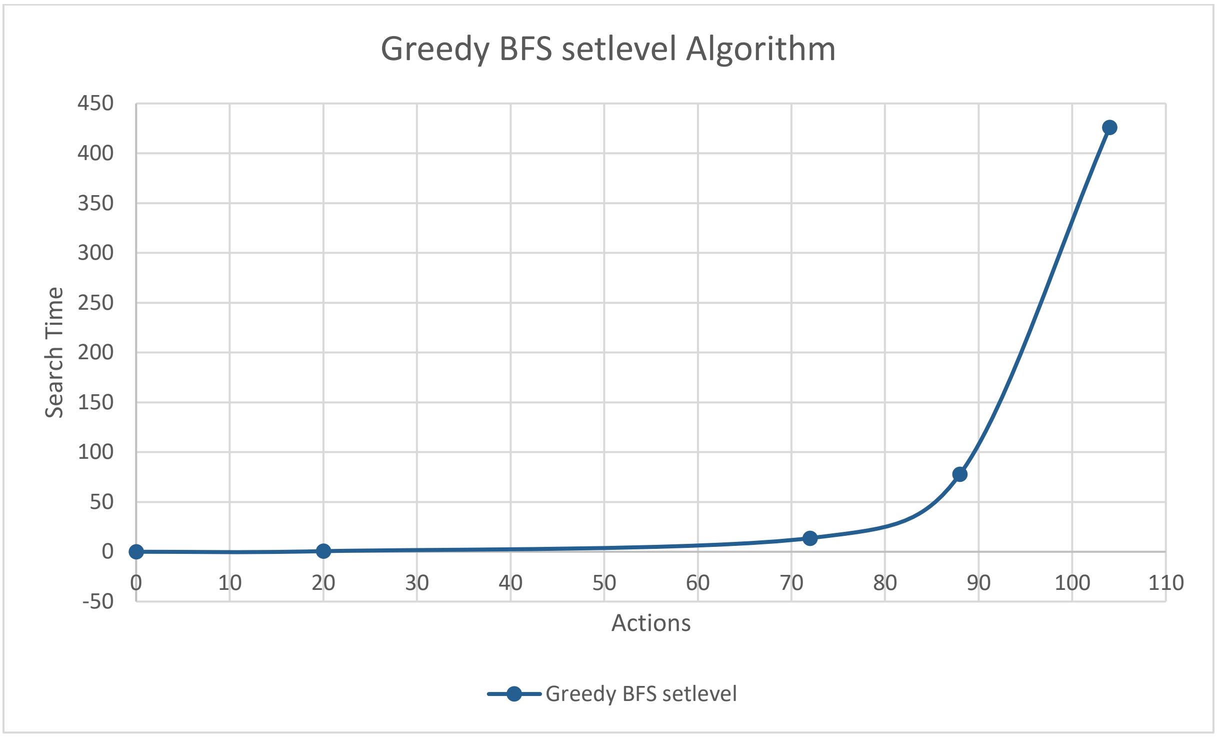


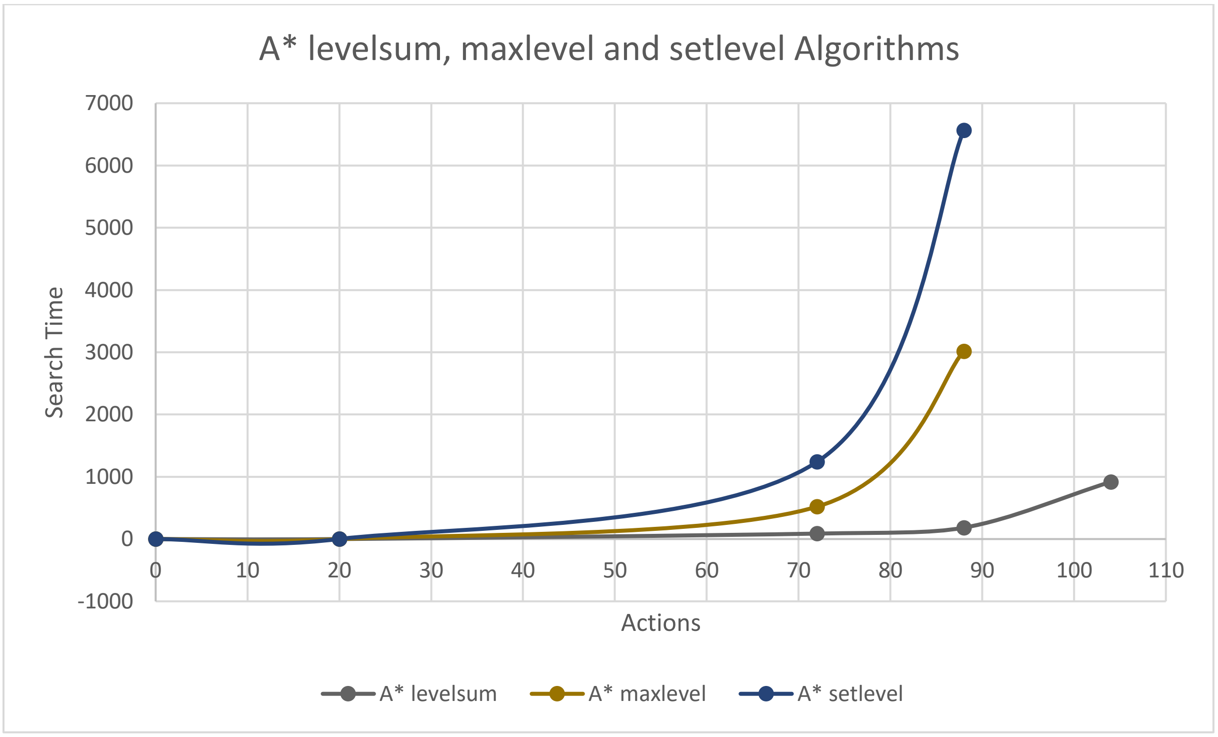


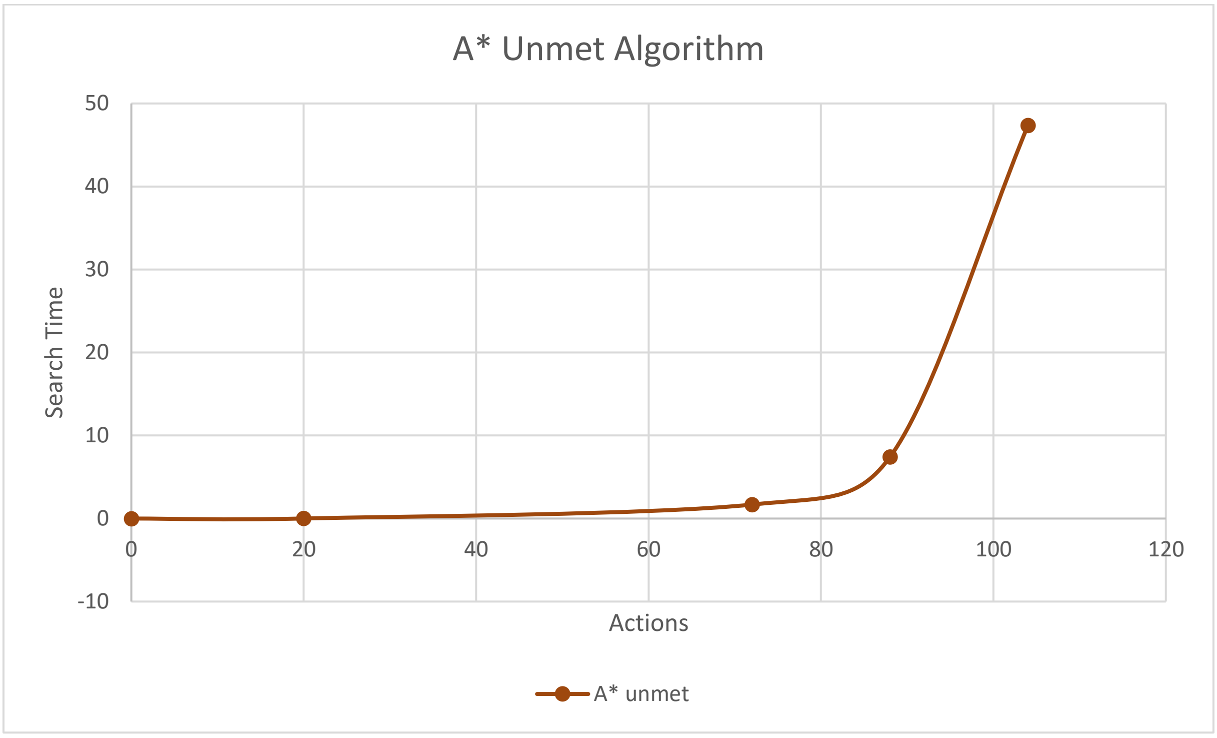
Greedy BFS Unmet Algorithm
Search Time
Actions
0,06
0,05
0,04
0,03
0,02
0,01
0
-0,01
0
10
20
30
40
50
60
70
80
90
100
110
Greedy BFS unmet



Greedy BFS levelsum & maxlevel Algorithm
Search Time
Actions
30
25
20
15
10
5
0
-5
0
20
40
60
80
100
120
Greedy BFS levelsum
Greedy BFS maxlevel


Greedy BFS setlevel Algorithm
Search Time
450
400
350
300
250
200
150
100
50
0
-50
0
10
20
30
40
50
60
70
80
90
100
110
Actions
Greedy BFS setlevel

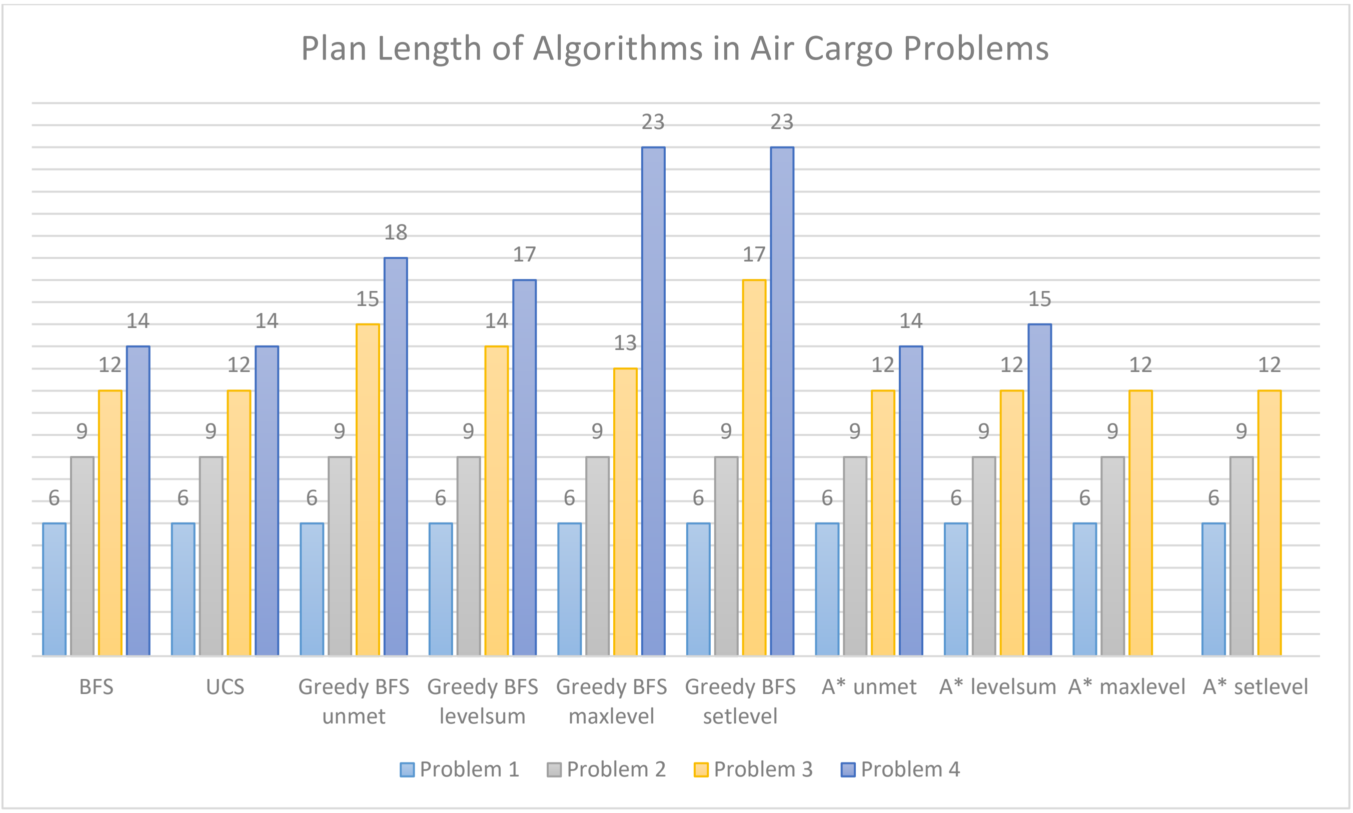

A* levelsum, maxlevel and setlevel Algorithms
Search Time
7000
6000
5000
4000
3000
2000
1000
0
-1000
0
10
20
30
40
50
60
70
80
90
100
110
Actions
A* levelsum
A* maxlevel
A* setlevel

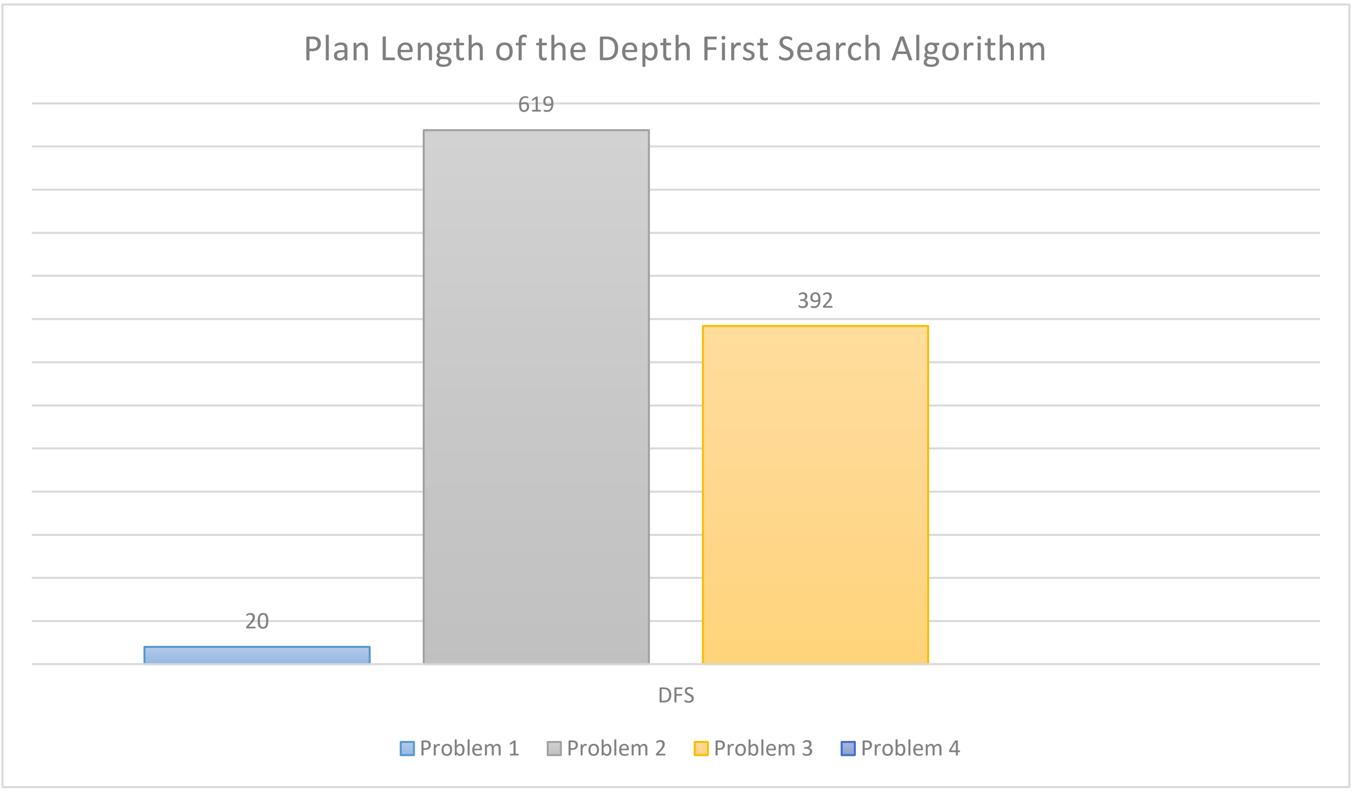


For the search time data, the fastest algorithm was the Greedy BFS unmet algorithm, for all four air cargo problems it had an average search time of 0.024323, the Depth First Search algorithm, could not complete the fourth problem so the search time would be infinity. The BFS and UCS uninformed search algorithms closely resemble the number of nodes graph for the BFS and UCS algorithms, with BFS being slightly better. The Greedy BFS levelsum and maxlevel algorithms were pretty much equal on time in the third problem but the levelsum algorithm then beat the maxlevel algorithm by nine seconds in the fourth problem.

For the A* algorithms, the unmet algorithm was the second fastest out of all the algorithms, as the Depth First Search algorithm could not solve problem four. The other A* heuristic algorithms (levelsum, maxlevel and setlevel) were all very slow with setlevel having the highest search time with 6563 seconds.

Length of plans returned by each algorithm on all search problems





Questions

1.

Which algorithm or algorithms would be most appropriate for planning in a very restricted domain (i.e., one that has only a few actions) and needs to operate in real time?

The best algorithms for planning in a very restricted domain with operation in real time, would be the uninformed searches BFS, DFS and UCS, as they show a very low number of nodes expanded as long as the actions are low. They also show a very low search time with 0.0050, 0.027 and 0.0080 seconds respectively. Another algorithm which would be great for low actions and fast search times would be the Greedy BFS unmet algorithm, as it has a search time of 0.0013 seconds and a very low number of expanded nodes of 29 for 20 actions.

2.

Which algorithm or algorithms would be most appropriate for planning in very large domains (e.g., planning delivery routes for all UPS drivers in the U.S. on a given day?)

For very large planning problems the Greedy BFS unmet, levelsum, maxlevel and setlevel algorithms are the best as these have good search times and have the least number of nodes expanded out of all the algorithms for the Air Cargo problem 4 with 104 actions.

3.

Which algorithm or algorithms would be most appropriate for planning problems where it is important to find only optimal plans?

For optimization we can't use the Greedy BFS algorithms as they lack optimization of the routes. The A* search algorithm with heuristics levelsum, maxlevel and setlevel are the best for finding optimal plans. This is because it keeps searching until it finds the best route to the goal which is different in Greedy BFS, as Greedy BFS eventually accepts a suboptimal solution to benefit the runtime of the search.