CS312 Lab 2 Report

Soumya Srividhya, 170010038
Anudeep Tubati, 170010039

I. OVERVIEW

This report summarises the observations and deductions from the simulations of various informed search algorithms on the Maze domain.

II. DESCRIPTION OF DOMAIN

a. State Space

The state space is the coordinate system made up of 1x1 squares. The agent can move either Down, Up, Right or Left from its current position. Additionally, the agent can move onto squares which are free (no boundary) only.

b. Start Node and Goal Node

The agent starts at (0, 0) every time and hence (0, 0) is the Start node. The Goal node can be anywhere, provided a path from (0, 0) using moving only Down, Up, Right or Left exists.

c. MoveGen and GoalTest

MoveGen(N, d)

```
nextStates <= ()
for neighbor n of state N within depth d do
        if n is not boundary and n not in (open ∪ closed) then
                nextStates.append(n)
return nextStates
```

GoalTest(N)

```
if N.data == '*' then
        return true
else
        return false
```

III. HEURISTIC FUNCTIONS

Two heuristic functions, namely, the Manhattan distance and the Chebyshev distance are used.

Manhattan Distance $((x_1, y_1), (x_2, y_2)) = |x_1 - x_2| + |y_1 - y_2|$

Chebyshev Distance $((x_1, y_1), (x_2, y_2)) = \max(|x_1 - x_2|, |y_1 - y_2|)$

IV. BEST FIRST SEARCH

It is observed that in most of the cases, both the heuristic functions give similar results. However, it is observed that Chebyshev performs worse sometimes when the difference of 2 components of distance is high. This is because it considers only the max component as the heuristic and ignores the smaller component. This makes the agent explore every state till the smaller component matches the bigger one. The results are tabulated below

Maze	States Explored		Distance to Goal	
	Manhattan	Chebyshev	x	y
maze5x5_1	34	88	13	15
maze5x5_2	79	63	1	7
maze6x6_1	127	169	16	18
maze6x6_2	80	96	1	13
maze7x15_1	313	299	43	21
maze7x15_2	62	55	5	19
maze15x7_1	689	439	19	75
maze15x7_2	707	643	2	66

Note: From now, if the heuristic function isn't mentioned, it is Manhattan by default.

V. BEST FIRST SEARCH VS. HILL CLIMBING

Maze	Best First Search			Hill Climbing		
	Time	Space (States Explored/Depth to Goal)	Optimal Soln.	Time	Space (States Explored/Depth to Goal)	Optimal Soln.
maze4x4_1	1.76e-04	1	Yes	3.5e-05	1	Yes
maze4x4_2	1.34e-04	1.875	Yes	-	-	No Soln.
maze5x5_1	8.7e-05	2.394	Yes	3.7e-05	1.15	Yes
maze5x5_2	2.8e-04	1.17	Yes	-	-	No Soln.
maze6x6_1	1.6e-04	1.90	No	-	-	No Soln.
maze6x6_2	3.42e-04	1.702	No	2.54e-04	1.17	No

It is evident from the observations that HC gives better results (space complexity O(1) and time complexity O(n)) when the heuristic function is monotonic in the path of start state and goal state. Otherwise, it fails to give a solution. However, Best First Search gives a solution (may not be optimal) all the time, at the cost of more space and time.

VI. BEAM WIDTH ANALYSIS

Maze	Beam Width	Time Taken (micro seconds)	Space (States Explored/Depth to Goal)
maze6x6_1	8	918	123/65
	40	465	127/65
	80	696	127/65
maze6x6_2	8	550	76/47
	40	147	80/47
	80	434	80/47
maze7x15_1	8	-	-
	40	795	313/133
	80	893	313/133
maze7x15_2	8	407	62/45
	40	119	62/45

	80	204	62/45
maze15x7_1	8	928	629/217
	40	1279	685/217
	80	1715	696/217
maze15x7_2	8	-	-
	40	1321	699/231
	80	1197	711/231

Since β=40 produces results for all the mazes and configurations and also performs better than β=80. Hence, β=40 is chosen as the optimal beam width.

VII. TABU SEARCH FOR DIFFERENT VALUES OF TABU TENURE

i) Maze 5x5

Manhattan Heuristic Function

TABU TENURE	TIME TAKEN (micro seconds)	Number of states explored
1	-	-
2	-	-
3	913	32
4	1647	34

From Tabu Tenure 5 to 15 there is no solution

Chebyshev Heuristic Function

TABU TENURE	Number of states explored	TIME TAKEN
1	-	-
2	-	-
3	-	-

4	63	0.00366
5	68	0.003337
6	66	0.00491
7	51	0.003172
8	51	0.00623
9	67	0.006845
10	53	0.006959
11	67	0.005641
12	68	0.008215
13	52	0.008534
14	52	0.007387
15	70	0.007848

ii) Maze 6x6

Manhattan Heuristic Function

TABU TENURE	TIME TAKEN	Number of states explored
1	-	-
2	0.001376	71
12	0.005616	65
13	0.005395	67
14	0.006916	65
15	0.007014	65

From tabu tenure 3 to 11 there is no solution

Chebyshev Heuristic Function

TABU TENURE	TIME TAKEN	Number of states explored
5	0.004093	62

Excluding tabu tenure 5 all other tabu tenure from 1 to 15 have no solution

iii) Mazes 7x15 and 15x7 didn't have any solutions

Tabu Tenure 5 seems to perform better than the rest (2/2 test cases using Chebyshev).

VIII. COMPARISON OF VND, BEAM SEARCH AND TABU SEARCH

Using Beam Width as 40 and Tabu Tenure to be 5 (with Chebyshev Heuristic)as they were found to be optimal earlier.

Maze	VND			Beam Search			Tabu Search		
	Time Taken (micro secs)	Space (States Explored/ Depth of Goal)	Optimal Soln.	Time Taken (micro secs)	Space (States Explored/ Depth of Goal)	Optimal Soln.	Time Taken (micro secs)	Space (States Explored/ Depth of Goal)	Optimal Soln.
maze 5x5_1	370	1.15	Yes	453	2.34	Yes	-	-	-
maze5 x5_2	534	1.22	Yes	326	1.172	Yes	1219	2.344	Yes
maze 6x6_1	617	2.14	No	488	1.90	No	-	-	-
maze 6x6_2	242	1.12	No	175	1.632	No	141	1.127	No
maze7 x15_1	592	2.23	No	602	2.32	No	-	-	-
maze1 5x7_1	1398	2.97	No	1413	3.01	No	-	-	-

IX. CONCLUSION

Tabu Search seems to have a lacklustre performance as compared to the other informed search algorithms. The reason being, not all search algorithms do well in all domains. In some domains, Tabu Search may outperform other algorithms (presumably domains where

perturbative approaches are more profound). For the maze domain, as a perturbative neighbour procurement isn't really possible, Tabu Search has a lacklustre performance.

Though Best First Search promises a solution everytime and has a close to optimal solution, its space complexity is also decently high as it maintains an Open list. Whereas, VND and Beam Search perform well on all the mazes.

VND performs well when the heuristic function is smoother (i.e., lesser local optimums from start state to goal state). It loses its performance if there are too many local optimums as it'll have to make the moveGen function denser each time it hits a local optimum. However, it guarantees a solution as it keeps increasing the density of the moveGen function and it'll hit the goal state at some depth.

Beam Search, though it performs better than VND, doesn't guarantee a solution as the number of nodes it takes into account simply might not be sufficient. VND also performs better in bigger graphs because it can quickly progress through the graph with less dense moveGen functions and upgrade when it gets stuck. This option isn't available in the case of Beam Search hence it wastes space and time by taking a fixed amount of nodes into consideration initially.