CS312: Artificial Intelligence Laboratory
A* Algorithm

Soumya Srividhya Doma, 170010038
Anudeep Tubati, 170010039

OVERVIEW

This report elaborates on the various heuristics used to implement the A* algorithm and their analysis.

I. 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 do
        if n is not boundary then
                nextStates.append(n)
return nextStates
```

GoalTest(N)

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

II. HEURISTIC FUNCTIONS USED

a. Overestimating Heuristic (Square of Euclidean Distance)

overestimate(node):
    return $(node.x - goal.x)^2 + (node.y - goal.y)^2$

b. Underestimating Heuristic (Manhattan Distance)

underestimate(node):
    return |node.x - goal.x| + |node.y - goal.y|

c. Monotone (Euclidean Distance)

monotone(node):
    return $\sqrt{(node.x - goal.x)2 + (node.y - goal.y)2}$

III. A* ALGORITHM ANALYSIS AND OBSERVATION

		Overestimate	Underestimate	Monotone
Maze 6x6	Time Taken	0.000756	0.000701	0.000565
	Path Length	67	65	65
Maze 15x7	Time Taken	0.002473	0.003143	0.002816
	Path Length	261	231	231
Maze 12x15	Time Taken	0.000839	0.000419	0.000317
	Path Length	102	96	96
Maze 13x14	Time Taken	0.001267	0.000850	0.000579
	Path Length	164	156	156
Maze 15x10	Time Taken	0.003088	0.003304	0.001622
	Path Length	215	177	177

As we can observe from the table, a Monotone heuristic gives the best performance and optimal results. Note that though an underestimating heuristic also gives optimal results, its performance is worse than the Monotone heuristic in terms of space & time complexity. This owes to the fact that Monotone never invokes *PropogateImprovement* and avoids the

recursive overhead of updating costs, as a node added to Closed list by a Monotone heuristic already has an optimal path. On the other hand, the overestimating heuristic finds suboptimal paths as it overestimates costs of nodes and may miss an optimal path because of the same reason.