POLITECNICO DI TORINO

DEPARTMENT OF CONTROL & COMPUTER ENGINEERING

COMPUTATIONAL INTELLIGENCE

LAB I

SET COVERING USING A* ALGORITHM

Name	Matricula	GitHub	Submission Date
Hossein Khodadadi	313884	Link_1	24/10/2023
Abolfazl Javidian	314441	Link_2	24/10/2023

Fall 2023/2024

1. Report: Analysis of A* Algorithm for Set-Covering Problem

Build a new H, consider the old H (distance), what about the special sets what about the order of sets?

There are two sets that can be considered special, a subset that covers all the U in that case the program stops the exploration (we can handle it if the distance is equal to zero), and a null set which cannot be generated using the code we provided. Since the states will be pushed to the queue based on their order, it may change the number of steps that A* algorithm takes to converge. But the optimal solution will be derived anyway.

1.1 Main Contribution and Changes

In this report, we analyzed an implementation of the A* algorithm for solving the set-covering problem. The following snippet code (Figure 1) is the main contribution of our work.

```
25  def distance(state):
26      return PROBLEM_SIZE - sum(
27          reduce(
28              np.logical_or,
29              [SETS[i] for i in state.taken],
30              np.array([False for _ in range(PROBLEM_SIZE)]),
31          )
32      )
33
34
35  def g_function(state):
36      return len(state.taken)
37
38
39  def h_function(state):
40      return distance(state)
41
42
43  def costFunction(state):
44      return g_function(state) + h_function(state)
```

Figure 1

1.2 Function Analysis:

distance(state): The distance function calculates the heuristic distance by determining the number of elements not covered by the selected sets. It serves as the heuristic for the A* algorithm.

g_function(state): The g_function calculates the cost from the initial state to the current state by computing the number of sets selected.

h_function(state): The h_function estimates the cost from the current state to the goal state, utilizing the distance function as a heuristic to approximate the remaining number of uncovered elements.

costFunction(state): The costFunction function combines the g_function and the h_function to calculate the total cost, which is used to determine the priority in the priority queue of the A* algorithm.

1.3 Results

We have repeated the experiment 100 times to calculate the average number of steps for the algorithm to converge. Results are as follows based on *Iteratory.py* file.

Algorithm	***# Iteration****	***Avg_Node_Exploration***	***Problem_Size***	***# Subsets***
A*	100	5	15	15