TDDC17 Lab2 : Search

# Part 1

See the code in the depository

# Part 2

1. In the vacuum cleaner domain in part 1, what were the states and actions? What is the branching factor?

The branching factor is 4. The branching factor is the maximum number of branches from a node.

The states are the positions in the environment: the grid. We have the start state, the goal state, when a wall is found and when a dirt is found.

The actions are: moving action (4 directions) and cleaning the dirt

2. What is the difference between Breadth First Search and Uniform Cost Search in a domain where the cost of each action is 1?

There is no difference between BFS and UCS. The only thing that changes is the implementation, BFS uses a FIFO queue and UCS uses a priority queue.

3. Suppose that h1 and h2 are admissible heuristics (used in for example A\*). Which of the following are also admissible?

a) (h1+h2)/2 is admissible

b) 2h1 is not always admissible

c) max (h1,h2) is admissible

4. If one uses A\* to search for a path to one specific square in the vacuum domain, what could the heuristic function (h) be? What could the cost function (g) be? Is your choice of (h) an admissible heuristic? Explain why.

The heuristic function could be the distance between 2 points. H = sqrt[delta(x)\*delta(y)] where delta(x) and delta(y) represents the absolute distance.

The function g could be the distance traveled by the agent.

H is an admissible heuristic in the given domain because it is impossible for the agent to find a shorter path than the one calculated by H.

5. Draw and explain. Choose your three favorite search algorithms and apply them to any problem domain (it might be a good idea to use a domain where you can identify a good heuristic function). Draw the search tree for them, and explain how they proceed in the searching. Also, include the memory usage. You can attach a hand-made drawing.

BFS : A – B – C – D – E – F – F – G – H – I – J – K

DFS : A – B – F – H – K

A\* : A – B – F – H – K

6. Look at all the offline search algorithms presented in chapter 3 plus A\* search. Are they complete? Are they optimal? Explain why!

BFS is complete if the branch factor is a finite number so that the algorithm can do backtracking and it will never get stuck in a node and optimal if the cost function is a non-decreasing function

DFS is not optimal or complete because it can get stuck if the branch factor is infinite.

Uniform cost search is a general form of the breath first search.

A\* is optimal is the heuristic function is admissible and complete if the branch is a finite number.

Iterative deepening search is complete if the branch is finite number and is optimal fi the heuristic function is admissible.

Bidirectional search completeness depends on BFS and DFS. It is not optimal because it depends also on BFS (optimal) and DFS(not optimal)

7. Assume that you had to go back and do Lab 1/Task 2 once more (if you did not use search already). Remember that the agent did not have perfect knowledge of the environment but had to explore it incrementally. Which of the search algorithms you have learned would be most suited in this situation to guide the agent's execution? What would you search for? Give an example.

We use BFD because the result was better than DFS. A\* search is difficult because we don’t know anything about the environment around the agent. So, the heuristic function is difficult to find in an unknown environment. Our objective should be to discover unexplored nodes.