Report LINGI2261: Assignment 2

Group N°...

Student1: .....

Student2: .....

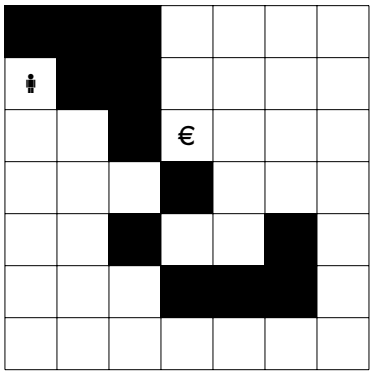
March 1, 2022

1 Search Algorithms and their relations (3 pts)

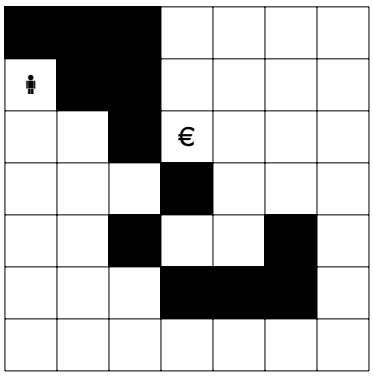
Consider the maze problems given on Figure 1. The goal is to find a path from 🚹 to € moving up, down, left or right. The black cells represent walls. This question must be answered by hand and doesn't require any programming.

1. Give a consistent heuristic for this problem. Prove that it is consistent. Also prove that it is admissible. **(1 pt)**

2. Show on the left maze the states (board positions) that are visited when performing a uniform-cost graph search, by writing the order numbers in the relevant cells. We assume that when different states in the fringe have the smallest value, the algorithm chooses the state with the smallest coordinate (i, j) ($(0, 0)$ being the bottom left position, i being the horizontal index and j the vertical one) using a lexicographical order. **(1 pt)**



3. Show on the right maze the board positions visited by A^* graph search with a manhattan distance heuristic (ignoring walls), by writing the order numbers in the relevant cells. A state is visited when it is selected in the fringe and expanded. When several states have the smallest path cost, they are visited in the same lexicographical order as the one used for uniform-cost graph search. **(1 pt)**



2 PageCollect problem (17 pts)

1. Model the PageCollect problem as a search problem; describe: **(2 pts)**
   - States
   - Initial state
   - Actions / Transition model
   - Goal test
   - Path cost function

2. Give an upper bound on the number of different states for a PageCollect problem with a map of size $n \times m$, with k pages to collect. Justify your answer precisely. **(1 pt)**

3. Give an admissible heuristic for a PageCollect instance with k pages. Prove that it is admissible. What is its complexity ? **(2 pts)**

5. **Implement** your solver. Extend the *Problem* class and implement the necessary methods and other class(es) if necessary. **(1 pt)**

6. **Experiment**, compare and analyze informed (*astar_search*) and uninformed (*breadth_first_graph_search*) graph search of aima-python3 on the 10 instances of PageCollect provided. Report in a table the time, the number of explored nodes and the number of steps to reach the solution. Are the number of explored nodes always smaller with *astar_search*? What about the computation time? Why? When no solution can be found by a strategy in a reasonable time (say **3 min**), indicate the reason (time-out and/or exceeded the memory). **(4 pts for the whole question)**

Inst.	A^* Graph			BFS Graph		
	NS	T(s)	EN	NS	T(s)	EN
i01						
i02						
i03						
i04						
i05						
i06						
i07						
i08						
i09						
i10						

NS: Number of steps — **T**: Time — **EN**: Explored nodes

6. **Submit** your program on INGInious, using the A^* algorithm with your best heuristic(s). Your file must be named *pagecollect.py*. You program must print to the standard output a solution to the PageCollect instance given in argument, satisfying the described output format. **(5 pts)**