

Report LINGI2261: Assignment 2



Group N°...

Student1:

Student2:


February 26, 2021

1 Search Algorithms and their relations (3 pts)

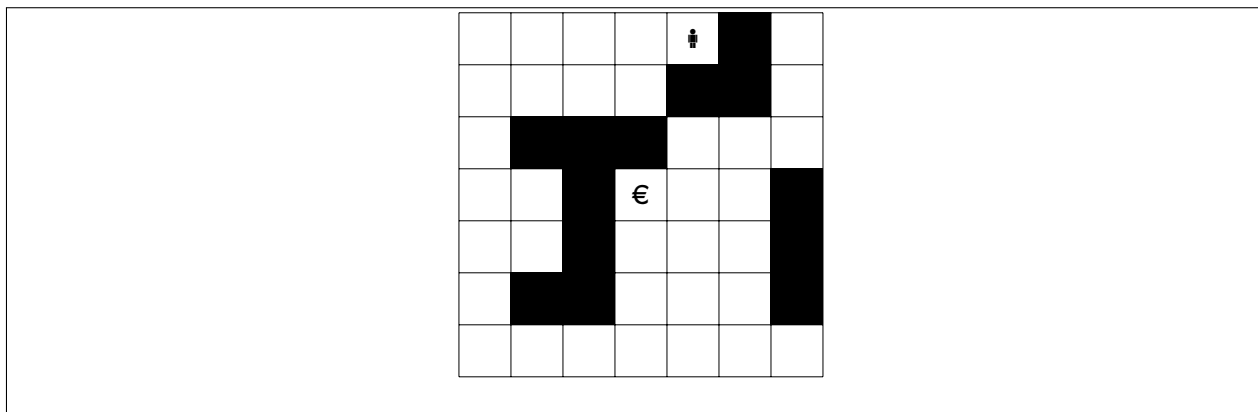
Consider the maze problems given on Figure 1. The goal is to find a path from  to  moving up, down, left or right. The black cells represent walls. This question must be answered by hand and doesn't require any programming.

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 Blocks planning problem (17 pts)

1. Model the Blocks planning problem as a search problem; describe: (2 pts)

- States
- Initial state
- Actions / Transition model
- Goal test
- Path cost function

2. Consider the following state for the a01 instance:

```
#####  
#           #  
#  c       #  
#  a       #  
####      #  
####      #  
##  b     #  
#####
```

According to the goal state, such a situation cannot lead to a solution. Can you find other similar situations (in general, not only on that specific instance) that leads to a deadlock? If so, describe two. (2 pts)

3. Why is it important to identify dead states? How are you going to take it into account in your solver? (2 pts)

4. **Describe** a possible (non trivial) heuristic to reach a goal state. Is your heuristic admissible and/or consistent? Why ? (2 pts)

5. **Implement** this problem. Extend the *Problem* class and implement the necessary methods and other class(es) if necessary. **Experiment**, compare and analyze informed (*astar_graph_search*) and uninformed (*breadth_first_graph_search*) graph searches of aima-python3 on the 10 instances of Blocks planning provided. Report in a table the time, the number of explored nodes, the number of remaining nodes in the queue and the number of steps to reach each solution. When no solution can be found by a strategy in a reasonable time (say **1 min**), indicate the reason (time-out and/or swap of the memory).

Are the number of explored nodes always smaller with *astar_graph_search*? What about the computation time? Why? (3 pts)

Inst.	A* Graph				BFS Graph			
	NS	T(s)	EN	RNQ	NS	T(s)	EN	RNQ
a01								
a02								
a03								
a04								
a05								
a06								
a07								
a08								
a09								
a10								

NS: Number of steps — T: Time — EN: Explored nodes — RNQ: Remaining nodes in the queue

6. **Submit** your program on INGIInious, using the *A** algorithm with your best heuristic(s). Your program must print to the standard output given a time limit of 1 minute, a solution to the Blocks planning instance passed as parameter to it, satisfying the described output format. Your program will be evaluated on 11 instances, one of which is hidden. We expect you to solve at least 8 out of the 11. (6 pts)

