## Report LINGI2261: Assignment 2

Group N°...

Student1: ......

Student2: ......

February 28, 2023

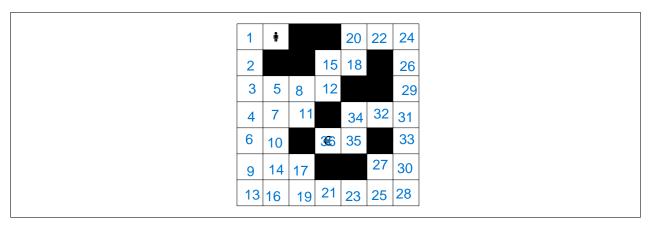
## 1 Search Algorithms and their relations (3 pts)

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.

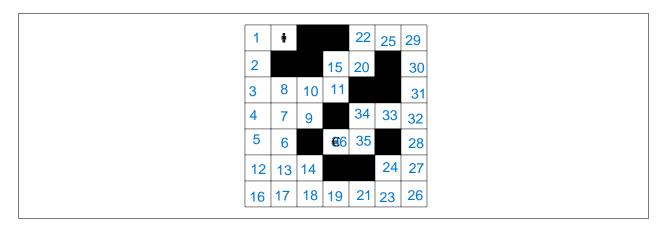
1. Give a consistent heuristic for this problem. Prove that it is consistent. Also prove that it is admissible. (1 pt)

Une heuristique consistante serait h(n) la distance de Manhattan entre les coordonnées de n jusqu' aux coordonnée l'objectif. Une heuristique est admissible si elle ne surestime jamais le coût pour atteindre le but. Ici c'est d'office le cas car on ne peut pas trouver de déplacement vers l'objectif plus court vu qu'on ignore la contrainte des murs. Cette heuristique est également consistante vu qu'il n'y a pas de chemin plus court que celui estimé par notre heuristique, il n'existe pas de chemin tel que le coût d'aller de n à n' par l'action a plus le coût estimé h(n') soit plus petit que h(n).

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 SoftFlow problem (17 pts)

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

Un état est une disposition de la grille et l'état initial est la disposition de la grille tel quel dans les instances, dans le programme cette grille est représenter par une liste. Les différentes actions possibles sont les déplacements d'une case de l'extrémité d'un câble non branché (représenter par les lettres) sur l'axe X ou Y si la case visé est libre (pas de mur # ni de câbles).

Chaque action crée une nouvelle disposition de la grille où l'extrémité s'est déplacé d'une case par rapport à l'état précédent et un chiffre remplace l'ancienne position de l'extrémité du câble (représentant le câble lui même).

Le goal test sera vrai si l'état respecte deux conditions: En 1, chaque câble du problème doit être relier à la sortie qui lui correspond et en 2, ce lien doit être le plus court possible.

Enfin, le coût d'un chemin est la somme des cases parcourus par les câbles, le déplacement d'une case à une autre est de 1.

s murs il pourrait y avoir k.((m.n)-(2k)	

2. Give an upper bound on the number of different states for a SoftFlow problem with a map of size

 $n \times m$ , with k cables to place. Justify your answer precisely. (1 pt)

Comme précédemment une heuristique admissible serait la distance de Manhattan entre les extrémités des câbles et leur sortie correspondante. Cette heuristique ne surestime jamais le coût du chemin puisqu'il n'existe pas de chemin plus court sur une grille que celui qui parcours la distance de Manhattan.

3. Give an admissible heuristic for a SoftFlow instance with k cables. Prove that it is admissible.

h(n) = abs((n.pos - n'.pos))

What is its complexity ? (2 pts)

n est l'extrémité du câble et n' sa sortie, .pos renvoi la position du point sur la grille. Cette fonction a une complexité en O(1)

- 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 SoftFlow 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)

			* Grap	h		S Gra		
	Inst.	NS	T(s)	EN	NS	T(s)	EN	
	i01							
	i02							
	i03							
	i04							
	i05							
	i06							
	i07							
	i08							
	i09							
	i10							
NS:	Number	of ste	ps — <sup>-</sup>	<b>T</b> : Tim	e — E	N: Ex	olored	nodes
		<u>'</u>						