TD 11: Algorithme $A^\star$

L'algorithme de Dijkstra (1959) permet de trouver tous les plus courts chemins dans un graphe pondéré (poids positifs) depuis un sommet source vers tous les sommets. L'algorithme de A^* (Peter Hart, Nils Nilsson and Bertram Raphael, 1968) calcule le plus court chemin entre une source et une unique destination de manière plus efficace, en changeant l'ordre dans lequel les sommets sont visités. Sa performance dépend grandement de l'heuristique utilisée pour déterminer dans quel ordre visiter les sommets.

L'algorithme A^* calcule le plus court chemin d'une origine o vers une destination d donnée. Pour cela, il utilise une heuristique H qui à chaque sommet u associe une estimation H_u de la distance qu'il reste à parcourir entre u et d. L'algorithme de Dijkstra et l'algorithme A^* sont très proches :

Entrées : Un graphe pondéré G, où le poids sur une arête (u, v) est $G_{u,v} \geq 0$. Un sommet o de G.
Sorties : Un vecteur D où D_u est la longueur du plus court chemin allant de s à u.

pour $u = 1$ ***à*** n **faire**
  $D_u \leftarrow +\infty$;
$Q = \{o\}$;
$D_o = 0$;
tant que Q *n'est pas vide* **faire**
  $u \leftarrow$ sommet de Q ayant le plus petit D_u ;
  Enlever u de Q;
  pour tous les v *voisins de* u **faire**
    si $D_u + G_{u,v} < D_v$ **alors**
      $D_v \leftarrow D_u + G_{u,v}$;
      $Q \leftarrow Q \cup \{v\}$;
Retourner D ;

Algorithme 1 : Dijkstra

Entrées : Un graphe pondéré G (où $G_{u,v} \geq 0$), une paire de sommets (o, d), une heuristique admissible H.
Sorties : La longueur du plus court chemin du sommet o au sommet d

pour $u = 1$ ***à*** n **faire**
  $D_u \leftarrow +\infty$;
$Q = \{o\}$;
$D_o = 0$;
tant que Q *n'est pas vide* **faire**
  $u \leftarrow$ sommet de Q ayant le plus petit $D_u + H_u$;
  si $u = d$ **alors**
    Retourner D_d;
  Enlever u de Q;
  pour tous les v *voisins de* u **faire**
    si $D_u + G_{u,v} < D_v$ **alors**
      $D_v \leftarrow D_u + G_{u,v}$;
      $Q \leftarrow Q \cup \{v\}$;
Retourner "pas de chemin entre o et d" ;

Algorithme 2 : $A^*(G, o, d, H)$

Exercice 1: Heuristique admissible

On dit qu'une **heuristique est admissible** si elle ne sur-estime jamais la distance qu'il reste à parcourir (pour tout sommet u, H_u est inférieur ou égal à la longueur du plus court chemin entre u et d).

1. On suppose que $H_u = 0$. Cette heuristique est-elle admissible quel que soit le graphe G ? Que donne l'algorithme A^* dans ce cas ?
2. Donner un exemple de graphe à quatre sommets et d'une heuristique non-admissible telle que l'algorithme A^* ne donne pas le plus court chemin.

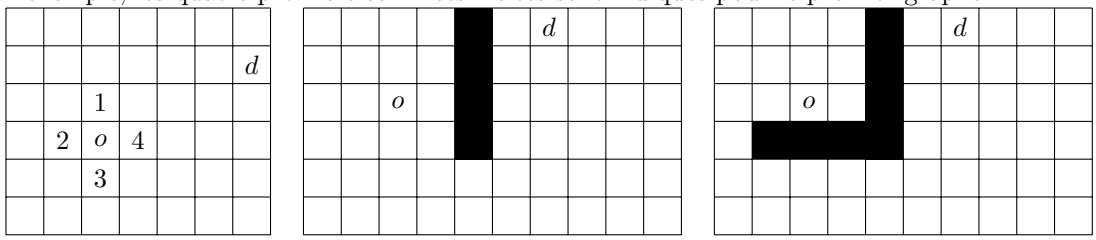
Exercice 2: A^* v.s. Dijkstra

On considère les graphes ci dessous. Chaque case blanche est un sommet. Chaque sommet est à distance 1 de ses voisins de gauche, droite, haut, bas (et n'est pas connecté aux autres). Pour un sommet u, on note (u_x, u_y) les coordonnées de ce sommet. On considère l'heuristique $H_u = |u_x - d_x| + |u_y - d_y|$.

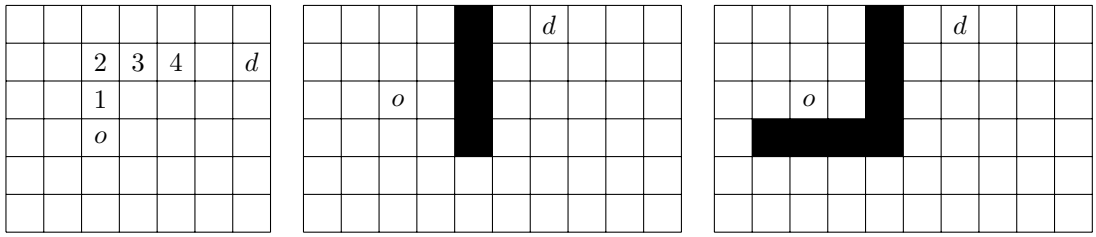
1. Cette heuristique est-elle admissible ?

2. Pour chacun des graphes ci dessous et les deux algorithmes Dijkstra et A^*, écrire un ordre possible de visite des sommets (on dit qu'un sommet est visité quand il est choisi dans l'ensemble Q). Pour exemple, les quatre premiers sommets visités sont indiqués pour le premier graphe.

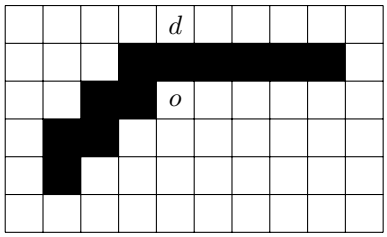
Dijkstra



A^*



3. On considère maintenant que l'on peut se déplacer en diagonale. Écrire les couples (D_u, H_u) pour chacun des sommets visités sur le graphe ci dessous.



Quels sont les sommets visités par l'algorithme A^* ?
Que se passe-t-il et pourquoi ?
Proposer une solution.

Exercice 3: Preuve de l'algorithme

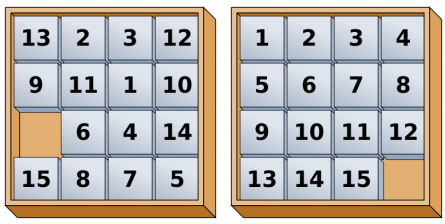
On suppose que l'heuristique est admissible. Soit ℓ^* la longueur du plus court chemin entre o et d. On note $(o, u_1, \ldots, u_k, d)$ un chemin de longueur ℓ^*. On raisonne par l'absurde et on suppose que lorsque l'algorithme termine, on a $D_d > \ell^*$. On note $(o, v_1, \ldots, v_k, d)$ le chemin qu'a trouvé l'algorithme A^*.

1. Montrer par récurrence que tous les sommets $u_1 \ldots, u_k$ auraient du être visités *avant*.
2. En déduire que si l'heuristique est admissible, alors l'algorithme A^* calcule bien la longueur du plus court chemin entre o et d.

Exercice 4: Taquin

On considère le problème du taquin ci-contre (le seul mouvement autorisé est de faire glisser une pièce dans le trou). On cherche à se ramener à la position initiale où tout est rangé. La distance est le nombre de mouvements qu'il faut effectuer. Les sommets du graphes sont les configurations possibles du jeu.
On considère les heuristiques :

1. $H^1_{configuration} = 15 -$ nombre de pièces bien placées.
2. $H^2_{configuration} = \sum_{i \in \text{pièces}}$ distance entre i et sa bonne position.



1. H^1 et/ou H^2 sont elles admissibles ?
2. On exécute l'algorithme A^* avec les deux heuristiques H^1 et H^2. Montrer que le nombre de sommets visités lorsque l'on fait tourner A^* avec H^2 est inférieur à celui lorsque que l'on fait tourner A^* avec H^1.