Adrien Paysant TP2 ASTAR IA

HE ARC 2020/21

A\*

Partie 1 : Heuristiques

*Etudier l’admissibilité des heuristiques suivantes :*

1. h0(n) = 0
2. h1(n) = “la distance entre n et B sur l’axe des x”
3. h2(n) = “la distance entre n et B sur l’axe des y”
4. h3(n) = “la distance à vol d’oiseau entre n et B”
5. h4(n) = “la distance de Manhattan entre n et B”
6. Si h0(n)=0 alors h0(n)0 (une distance est toujours positive ou nulle), de plus si h0(n)=0 alors h0(n)h\*(n) (une distance est toujours positive ou nulle).

En somme, 0h0(n) h\*(n), l’heuristique est donc admissible.

1. Il y a trois cas :
   1. Soit B et n sont alignés selon l’axe Ox;
   2. Soit B et n sont alignés selon l’axe Oy;
   3. Soit B et n ne sont alignés ni selon Ox, ni selon Oy.
2. Si B et n sont alignés selon l’axe Ox, alors h1(n) (une distance est toujours positive ou nulle), d’autre part dans ce cas h1(n)= h\*(n) qui est un cas particulier de h1(n)h\*(n).

En somme, 0h1(n) h\*(n), l’heuristique est donc admissible.

1. Si B et n sont alignés selon l’axe Oy, alors h1(n)= 0 qui est un cas particulier de h1(n)0.

Dans ce cas, h\*(n)h1(n)=0 (une distance est toujours positive ou nulle).

En somme, 0h1(n) h\*(n), l’heuristique est donc admissible.

1. Si B et n ne sont alignés ni selon Ox, ni selon Oy, alors d’après les propriétés des triangles, h\*(n) = h1(n) avec A le projeté orthogonal sur l’axe Ox de B.

De plus h1(n)(une distance est toujours positive ou nulle).

En somme, 0h1(n) h\*(n), l’heuristique est admissible.

L’heuristique étant toujours admissible dans chacun des cas, alors cela démontre que l’heuristique considérée est admissible.

1. Il y a trois cas :
   1. Soit B et n sont alignés selon l’axe Oy;
   2. Soit B et n sont alignés selon l’axe Ox;
   3. Soit B et n ne sont alignés ni selon Ox, ni selon Oy.
2. Si B et n sont alignés selon l’axe Oy, alors h2(n) (une distance est toujours positive ou nulle), d’autre part dans ce cas h2(n)= h\*(n) qui est un cas particulier de h2(n)h\*(n).

En somme, 0h1(n) h\*(n), l’heuristique est donc admissible.

1. Si B et n sont alignés selon l’axe Ox, alors h2(n)= 0 qui est un cas particulier de h1(n)0.

Dans ce cas, h\*(n)h2(n)=0 (une distance est toujours positive ou nulle).

En somme, 0h2(n) h\*(n), l’heuristique est donc admissible.

1. Si B et n ne sont alignés ni selon Ox, ni selon Oy, alors d’après les propriétés des triangles, h\*(n) = h2(n) avec A le projeté orthogonal sur l’axe Oy de B.

De plus h2(n)(une distance est toujours positive ou nulle).

En somme, 0h2(n) h\*(n), l’heuristique est admissible.

L’heuristique étant toujours admissible dans chacun des cas, alors cela démontre que l’heuristique considérée est admissible.

1. h3(n) (une distance est positive ou nulle), de plus la distance à vol d’oiseau est la plus courte distance reliant le point, donc h3(n)h\*(n). En somme, 0h3(n) h\*(n), l’heuristique est admissible.
2. L’heuristique n’est pas admissible, preuve par contre-exemple :

Si A(0,0) et B(3/2,2) alors h4([AB])=2+1.5=3.5 or les villes que nous considérons peuvent être reliées par des droite donc h\*([AB])==2.5

Or 2.5<3.5, donc l’heuristique est invalide.

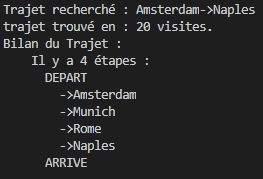
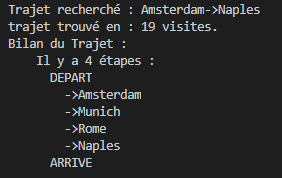
Partie 2 : Implémentation

* Le dossier data contient deux fichiers, l’un, *connections.txt* permet de référencer les liens entre les villes, le second fichier, *positions.txt*, permet d’avoir les informations sur les villes.
* Le fichier *CityLink.py* contient deux classes ainsi que des méthodes pour récupérer et mettre en forme les données du dossier data.
  + La classe City permet de représenter une ville ;
  + La classe Link représente une connexion entre deux villes ;
  + Les différentes méthodes permettent de lire les données et de les mettre en forme dans un dictionnaire grâce à la fonction *readAll*.
* Le fichier *main.py* contient la saisie des paramètres de recherche ainsi que l’appel à *readAll* pour obtenir les données ; ainsi que l’appel effectif de la recherche A\* et enfin une mise en forme de l’affichage du résultat de la recherche.
* Le fichier *aStar.py* contient l’implémentation de l’algorithme ainsi que la définition des différentes heuristiques.

Partie 3 : Expérimentation

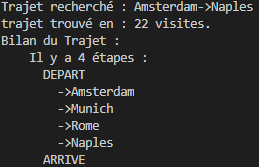
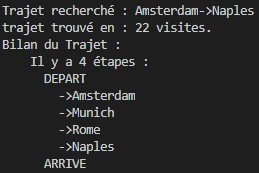
1. *L’usage d’heuristiques divers influe-t-elle sur l’efficacité de la recherche ?*

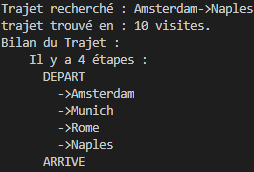
Oui, cette influence est démontré par l’exemple suivant :



A gauche, h0,

A droite h1.





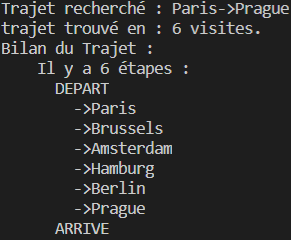
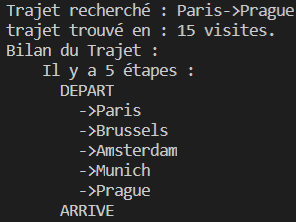
A gauche h2,

A droite h3,

En bas h5.

Cet exemple de trajet entre Amsterdam et Naples démontre bien que le choix de l’heuristique influe sur la performance de la recherche.

1. *Existe-t-il des exemples où le choix de l’heuristique change le chemin ?*

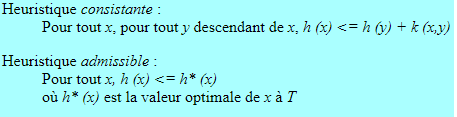
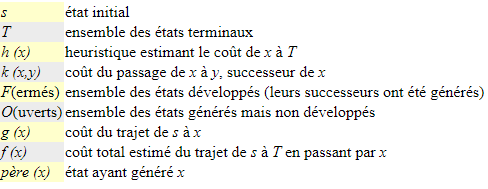
Oui, cela est le cas pour le trajet Paris->Prague qui différent selon entre h3 et h4. En effet, la recherche avec la distance de Manhattan rajoute une étape, comme le montrent les sorties ci-dessous.

1. *Dans un cas réel, quelle heuristique utiliseriez-vous ? Pourquoi ?*

L’heuristique 3, à vol d’oiseau me parait être la plus performante par rapport à ses consœurs, le vol d’oiseau étant toujours la plus petite distance entre deux points.

En effet, h0 n’apporte aucune plus-value à l’usage (heuristique nulle), h1 et h2 sont par leur définition inférieur ou égaux à h3 ; h4 n’est pas admissible, donc h3 est bien la meilleure approche à considérer.

1. *Aller plus loin : chercher la définition d’une heuristique monotone/consistante.*
   1. *Quel est son impact sur les performances de A\* ?*
   2. *Comment améliorer son code en conséquence ?*
   3. *Parmi les 5 heuristiques, y’en a-t-il des monotones ? Si non, proposer en une (sans implémentation) pour notre problème du voyageur.*
2. Une heuristique est dite consistante si :

Source : [Cours info IMT Atlantique](https://web.imt-atlantique.fr/x-info/pdavid/Enseignement/IA/poly-ia/resolution/a-etoile.html)

Cette définition est traduisible comme suit : la différence entre l'heuristique de deux nœuds quelconques reliés ne surestime pas la distance réelle entre ces nœuds.

L’impact est positif, une telle heuristique permettrait d’améliorer les performances de l’algorithme A\*.

1. Utiliser l’heuristique en question permet d’améliorer le code.
2. Trivialement, l’heuristique h0 est consistante car 0x . Rappelons qu’ici x est une distance, donc par définition x Donc on a bien h(x)h(y)+k(x,y)