Adrien Paysant

HE ARC 2020/21

TP2 ASTAR IA

A*

Partie 1 : Heuristiques

Etudier l'admissibilité des heuristiques suivantes :

a) $h_0(n) = 0$
b) $h_1(n)$ = "la distance entre n et B sur l'axe des x"
c) $h_2(n)$ = "la distance entre n et B sur l'axe des y"
d) $h_3(n)$ = "la distance à vol d'oiseau entre n et B"
e) $h_4(n)$ = "la distance de Manhattan entre n et B"

a) Si $h_0(n)=0$ alors $h_0(n)\geq 0$ (une distance est toujours positive ou nulle), de plus si $h_0(n)=0$ alors $h_0(n)\leq h^*(n)$ (une distance est toujours positive ou nulle).
En somme, $0\leq h_0(n) \leq h^*(n)$, l'heuristique est donc admissible.

b) Il y a trois cas :
   i. Soit B et n sont alignés selon l'axe O_x ;
   ii. Soit B et n sont alignés selon l'axe O_y ;
   iii. Soit B et n ne sont alignés ni selon O_x, ni selon O_y.

(i) Si B et n sont alignés selon l'axe O_x, alors $h_1(n)\geq 0$ (une distance est toujours positive ou nulle), d'autre part dans ce cas $h_1(n)= h^*(n)$ qui est un cas particulier de $h_1(n)\leq h^*(n)$.
En somme, $0\leq h_1(n) \leq h^*(n)$, l'heuristique est donc admissible.

(ii) Si B et n sont alignés selon l'axe O_y, alors $h_1(n)= 0$ qui est un cas particulier de $h_1(n)\leq 0$.

Dans ce cas, $h^*(n)\geq h_1(n)=0$ (une distance est toujours positive ou nulle).

En somme, $0\leq h_1(n) \leq h^*(n)$, l'heuristique est donc admissible.

(iii) Si B et n ne sont alignés ni selon O_x, ni selon O_y, alors d'après les propriétés des triangles, $h^*(n)\geq ||OA|| = h_1(n)$ avec A le projeté orthogonal sur l'axe O_x de B.
De plus $h_1(n)\geq 0$ (une distance est toujours positive ou nulle).

En somme, $0\leq h_1(n) \leq h^*(n)$, l'heuristique est admissible.

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

c) Il y a trois cas :
   i. Soit B et n sont alignés selon l'axe O_y ;
   ii. Soit B et n sont alignés selon l'axe O_x ;
   iii. Soit B et n ne sont alignés ni selon O_x, ni selon O_y.

(i) Si B et n sont alignés selon l'axe O_y, alors $h_2(n)\geq 0$ (une distance est toujours positive ou nulle), d'autre part dans ce cas $h_2(n)= h^*(n)$ qui est un cas particulier de $h_2(n)\leq h^*(n)$.
En somme, $0\leq h_1(n) \leq h^*(n)$, l'heuristique est donc admissible.

(ii) Si B et n sont alignés selon l'axe O_x , alors $h_2(n)= 0$ qui est un cas particulier de $h_1(n) \leq 0$.

Dans ce cas, $h^*(n) \geq h_2(n)=0$ (une distance est toujours positive ou nulle).

En somme, $0 \leq h_2(n) \leq h^*(n)$, l'heuristique est donc admissible.

(iii) Si B et n ne sont alignés ni selon O_x, ni selon O_y, alors d'après les propriétés des triangles, $h^*(n) \geq ||OA|| = h_2(n)$ avec A le projeté orthogonal sur l'axe O_y de B.

De plus $h_2(n) \geq 0$ (une distance est toujours positive ou nulle).

En somme, $0 \leq h_2(n) \leq h^*(n)$, l'heuristique est admissible.

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

d) $h_3(n) \geq 0$ (une distance est positive ou nulle), de plus la distance à vol d'oiseau est la plus courte distance reliant le point, donc $h_3(n) \leq h^*(n)$. En somme, $0 \leq h_3(n) \leq h^*(n)$, l'heuristique est admissible.

e) L'heuristique n'est pas admissible, preuve par contre-exemple :
Si A(0,0) et B(3/2,2) alors $h_4([AB])=2+1.5=3.5$ or les villes que nous considérons peuvent être reliées par des droite donc $h^*([AB])=\sqrt{\left(\frac{3}{2}-0\right)^2+(2-0)^2}=2.5$
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 :

```
Trajet recherché : Amsterdam->Naples
trajet trouvé en : 20 visites.
Bilan du Trajet :
    Il y a 4 étapes :
      DEPART
        ->Amsterdam
        ->Munich
        ->Rome
        ->Naples
      ARRIVE
```

A gauche, h0,
A droite h1.

```
Trajet recherché : Amsterdam->Naples
trajet trouvé en : 19 visites.
Bilan du Trajet :
    Il y a 4 étapes :
      DEPART
        ->Amsterdam
        ->Munich
        ->Rome
        ->Naples
      ARRIVE
```

```
Trajet recherché : Amsterdam->Naples
trajet trouvé en : 22 visites.
Bilan du Trajet :
    Il y a 4 étapes :
      DEPART
        ->Amsterdam
        ->Munich
        ->Rome
        ->Naples
      ARRIVE
```

A gauche h2,

A droite h3,

En bas h5.

```
Trajet recherché : Amsterdam->Naples
trajet trouvé en : 10 visites.
Bilan du Trajet :
    Il y a 4 étapes :
      DEPART
        ->Amsterdam
        ->Munich
        ->Rome
        ->Naples
      ARRIVE
```

```
Trajet recherché : Amsterdam->Naples
trajet trouvé en : 22 visites.
Bilan du Trajet :
    Il y a 4 étapes :
      DEPART
        ->Amsterdam
        ->Munich
        ->Rome
        ->Naples
      ARRIVE
```

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

2) *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.

```
Trajet recherché : Paris->Prague
trajet trouvé en : 6 visites.
Bilan du Trajet :
    Il y a 6 étapes :
      DEPART
        ->Paris
        ->Brussels
        ->Amsterdam
        ->Hamburg
        ->Berlin
        ->Prague
      ARRIVE
```

```
Trajet recherché : Paris->Prague
trajet trouvé en : 15 visites.
Bilan du Trajet :
    Il y a 5 étapes :
      DEPART
        ->Paris
        ->Brussels
        ->Amsterdam
        ->Munich
        ->Prague
      ARRIVE
```

3) *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.

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

a) Une heuristique est dite consistante si :

Heuristique *consistante* :
Pour tout x, pour tout y descendant de x, $h(x) <= h(y) + k(x,y)$

Heuristique *admissible* :
Pour tout x, $h(x) <= h^*(x)$
où $h^*(x)$ est la valeur optimale de x à T

T	ensemble des états terminaux
$h(x)$	heuristique estimant le coût de x à T
$k(x,y)$	coût du passage de x à y, successeur de x

Source : Cours info IMT Atlantique

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*.

b) Utiliser l'heuristique en question permet d'améliorer le code.

c) Trivialement, l'heuristique h0 est consistante car $0 \leq x \ \forall x \geq 0$. Rappelons qu'ici x est une distance, donc par définition $x \geq 0$. Donc on a bien $h(x) \leq h(y)+k(x,y)$