TD - Introduction à l'Intelligence Artificielle n° 7
(Correction)

Recherche heuristique dans les graphes d'états

Exercice 1 Tourisme en Roumanie

On considère un touriste en fin de vacances en Roumanie et qui cherche à rejoindre l'aéroport situé à Bucharest par le plus court chemin. On considère la carte (simplifiée) de la Roumanie de la figure 1 et on dispose par ailleurs d'informations supplémentaires résumées dans le tableau 1.

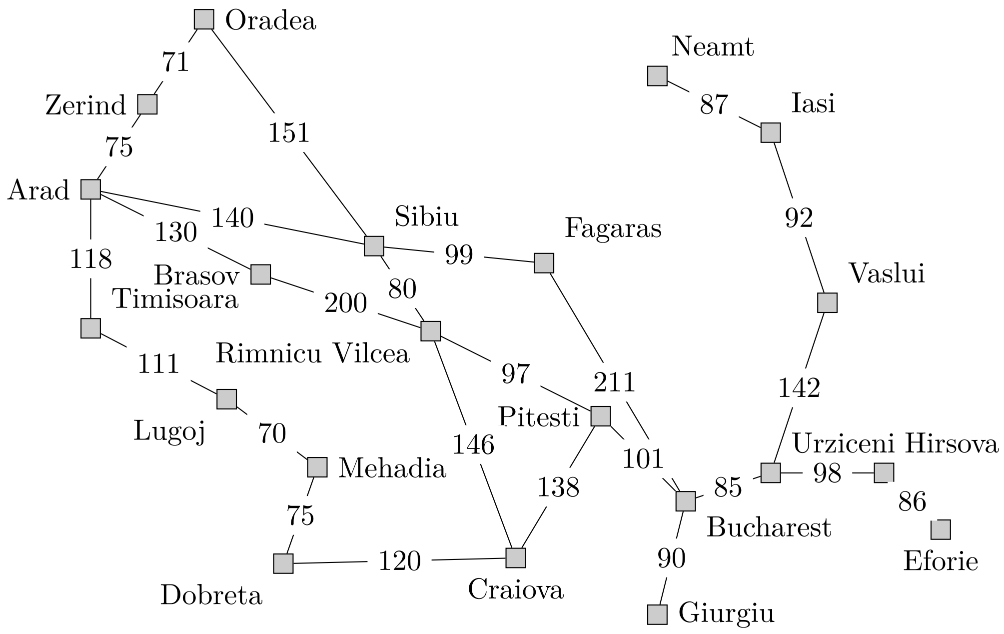


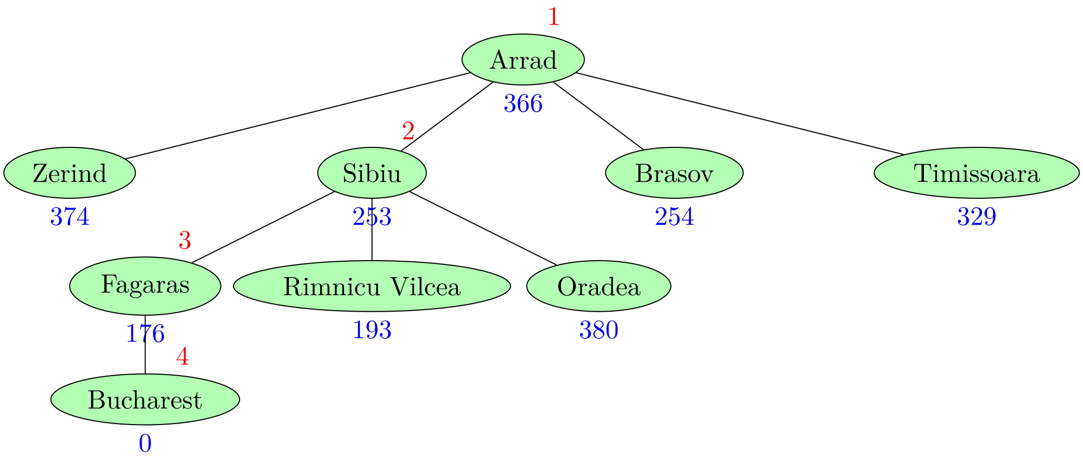
Figure 1 – Carte routière simplifiée de la Roumanie

1. Proposer une fonction heuristique h pour le problème du voyageur.

   Correction : La fonction heuristique h est en fait donnée sur la table, il s'agit de la distance en ligne droite entre la ville considérée et la ville destination. Cette heuristique est bien sûr admissible (car minorante ; A/A* est admissible si l'heuristique est minorante).

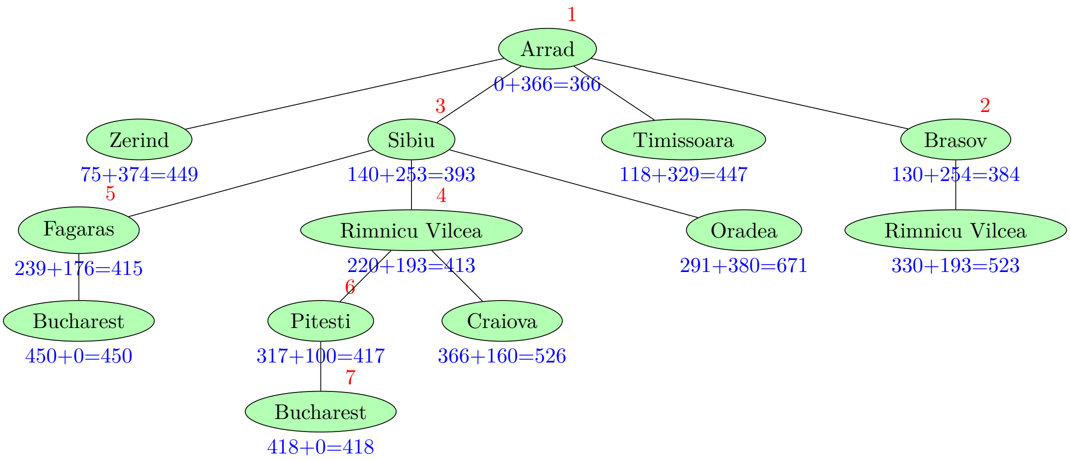
2. En vous aidant des informations figurant dans l'annexe ci-dessous, appliquer l'algorithme du *meilleur d'abord* pour aller de Arad à Bucharest.

   Correction : L'algorithme de recherche Meilleur d'abord classe les noeuds à développer suivant h. Remarquons que le chemin trouvé n'est pas le plus court (c'est celui passant par Riminicu et Pitesti).



3. Appliquer l'algorithme de recherche A* à ce problème avec la même fonction heuristique h. Quelle est la différence entre les deux recherches ?

   Correction :



4. Supposons que désormais, le coût d'un chemin est fonction du temps de parcours. Sachant que la vitesse moyenne sur les routes en Roumanie est de 60 km/h, un béotien proposerait de s'inspirer de l'heuristique précédente et donc de convertir la distance en durée. Par contre ce que ce béotien ignore est que l'on circule deux fois plus vite entre Fagaras et Bucharest grâce à l'autoroute, et que donc le coût réel du trajet entre ces deux villes est deux fois moindre. Qu'est-ce que cette modification de coût et d'heuristique change à la solution trouvée par l'algorithme A* ? La solution trouvée est-elle optimale et pourquoi ?

   Correction : Etant donné que les valeurs heuristiques ne changent pas (si on compte la durée en minute par exemple), le résultat de la recherche est inchangé. Par contre le chemin n'est plus optimal et cette fois-ci, le meilleur chemin est en passant par Fagaras. Le problème est que l'heuristique n'est plus admissible.

Annexe

Arad	366						
Brasov	254	Fagaras	176	Mehadia	241	Sibiu	253
Bucharest	0	Giurgiu	77	Neamt	234	Timisoara	329
Craiova	160	Hirsova	151	Oradea	380	Urzicen	80
Dobreta	242	Iasi	226	Pitesti	100	Vaslui	199
Eforie	161	Lugoj	244	Rimnicu Vilcea	193	Zerind	374

.

Table 1 – Distance à vol d'oiseau jusqu'à Bucharest

Exercice 2 Problème du taquin

On considère le problème classique du Taquin, avec comme état initial $\begin{array}{|ccc|}\hline 2 & 8 & 3 \\ 1 & 6 & 4 \\ 7 & & 5 \\ \hline\end{array}$ et comme état terminal $\begin{array}{|ccc|}\hline 1 & 2 & 3 \\ 8 & & 4 \\ 7 & 6 & 5 \\ \hline\end{array}$.

La figure 2 représente l'arbre de recherche construit par l'algortithme de recherche *en largeur d'abord*.

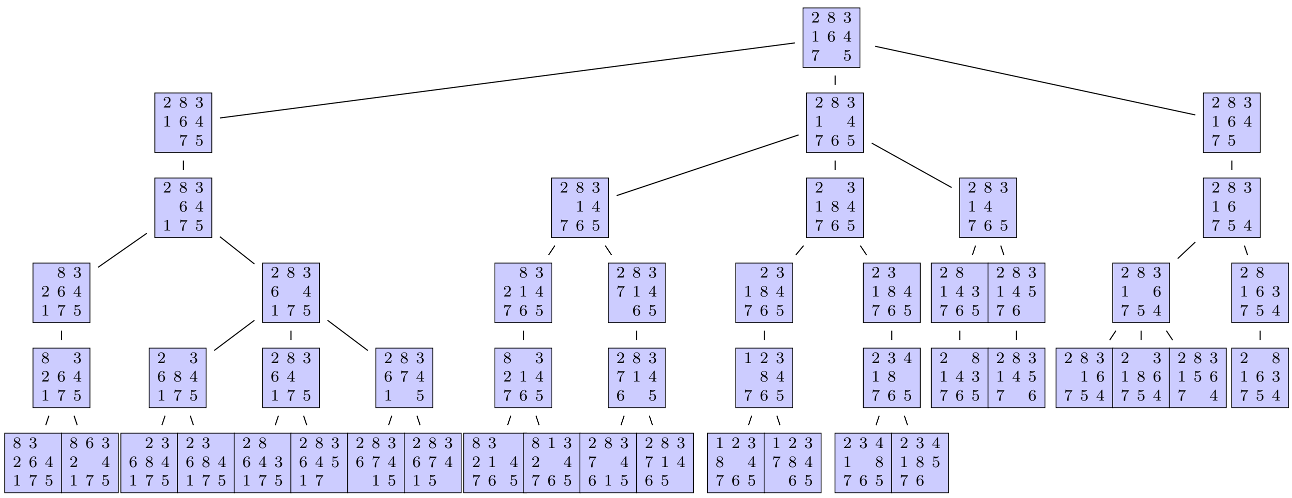


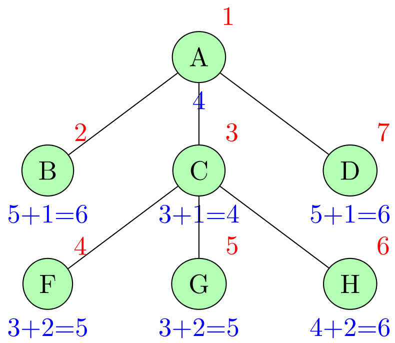
Figure 2 – Développement en largeur de l'arbre de recherche pour un problème de taquin

1. Proposer au moins 2 fonctions heuristiques pour le problème du Taquin.

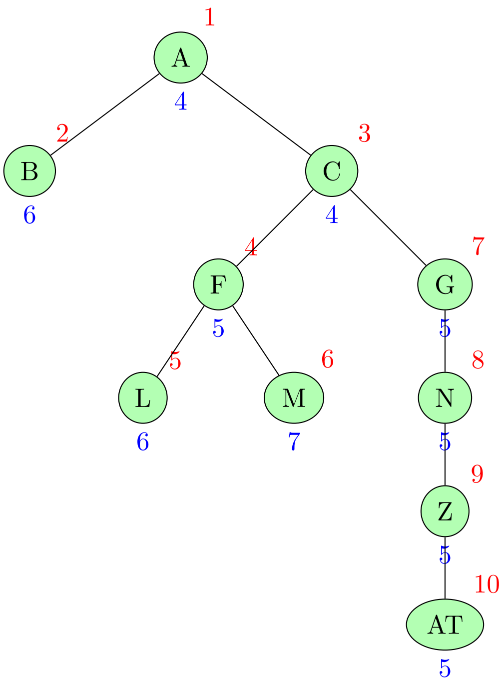
   Correction :
   h_1 : nombre de pieces mal placée
   h_2 : somme des distances de Manhattan pour chaque pièce par rapport à sa position finale.

2. Appliquer l'algorithme IDA^* avec chacune de ces deux heuristiques, en prenant comme fonction g (coût du chemin parcouru) le nombre de pièces déplacées depuis l'état initial.

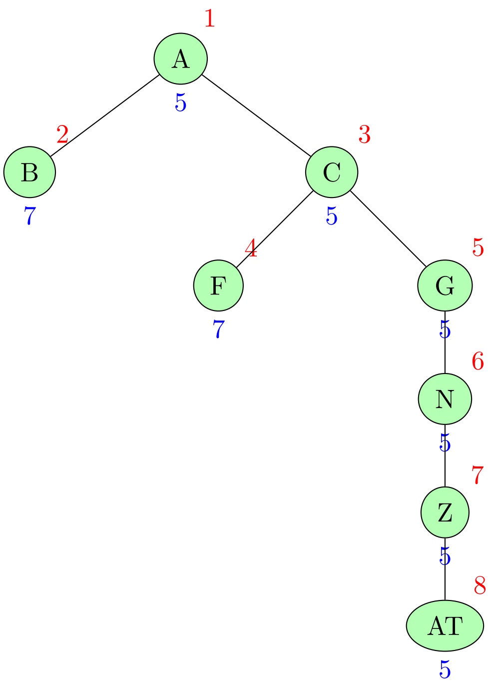
   Correction : Nommons les noeuds suivant l'ordre alphabétique et de gauche à droite.
   Borne initiale : $b = h_1(A) = 4$.



Retourne Echec(5).
$b = 5$



Retourne $\{A, C, G, N, Z, AT\}$.
Pour h_2 :
Borne initiale : $b = h_2(A) = 5$.



3. Comparer le nombre de noeuds développés avec ces 2 heuristiques. Ce résultat était-il prévisible

?

Correction : Avec h_2 la recherche est plus efficace. C'était en effet prévisible car quelque soit le noeud n, $h_2(n) \geq h_1(n)$. h_2 domine h_1. Tant que la propriété d'admissibilité est conservée, il est préférable d'utiliser l'heuristique *la plus forte*. D'ailleurs, s'il on a plusieurs heuristiques sans qu'on sache laquelle est dominante, il est toujours possible de prendre le max des heuristiques.

4. Formaliser les contraintes sur le déplacement des pièces. En levant progressivement ces contraintes, montrer que l'on peut retrouver les deux heuristiques précédentes.

Correction : Le principe est de lever les contraintes sur le problème pour générer automatiquement des heuristiques. Ces contraintes portent souvent sur les opérateurs. Pour le taquin : une pièce peut aller de A vers B si A et B sont adjacent et B est vide. La relaxation implique 3 possibilités :

(a) peut aller de A à B si A et B sont adjacent
(b) peut aller de A à B si B est vide.
(c) peut aller de A à B

La dernière par exemple, revient à inverser le contenu des cases A et B. La valeur exacte de la distance à la solution est alors h1. Et la première donne h2 . C'est avec cette méthode que fut construite la première heuristique "utile" pour le Rubix Cube.

5. Reprendre les questions 2 et 3 avec A^*.

Correction : Pour h_1 :

Frontière {n(f,g)}	**Déjà développés**	**Développé** n	**Fils** n'	$h_1(n')$	$g(n')$	$f = g + h$	**maj**
{A(4,0)}	$\emptyset$	A	B	5	1	6	
			C	3	1	4	
			D	5	1	6	
{C(4,1),B(6,1),D(6,1)}	{A}	C	F	3	2	5	
			G	3	2	5	
			H	4	2	6	
{F(5,2),G(5,2),H(6,2),B(6,1), D(6,1)}	{A,C}	F	L	3	3	6	
			M	4	3	7	
{G(5,2),L(6,3),H(6,2),B(6,1), D(6,1),M(7,3) }	{A,C,F}	G	N	2	3	5	
			0	4	3	7	
{N(5,3),L(6,3),H(6,2),B(6,1), D(6,1),M(7,3),O(7,3) }	{A,C,F,G}	N	Z	1	4	5	
{Z(5,4),L(6,3),H(6,2),B(6,1), D(6,1),M(7,3),O(7,3) }	{A,C,F,G,N}	Z	AT	0	5	5	
			AU	2	5	7	
{AT(5,5),L(6,3),H(6,2),B(6,1), D(6,1),AU(7,5),M(7,3),O(7,3) }	{A,C,F,G,N,Z}	AT					

Pour h_2 :

Frontière {n(f,g)}	**Déjà développés**	**Développé** n	**Fils** n'	$h_2(n')$	$g(n')$	$f = g + h$	**maj**
{A(5,0)}	$\emptyset$	A	B	6	1	7	
			C	4	1	5	
			D	6	1	7	
{C(5,1),B(7,1),D(7,1)}	{A}	C	F	5	2	7	
			G	3	2	5	
			H	5	2	7	
{G(5,2),F(7,2),H(7,2),B(7,1), D(7,1)}	{A,C}	G	N	2	3	5	
			P	4	3	7	
{N(5,3),P(7,3),F(7,2),H(7,2), B(7,1),D(7,1)}	{A,C,G}	N	Z	1	4	5	
{Z(5,3),P(7,3),F(7,2),H(7,2), B(7,1),D(7,1)}	{A,C,G,N}	Z	AT	0	5	5	
			AU	2	5	7	
{AT(5,5),P(7,3),F(7,2),H(7,2), B(7,1),AU(7,5),D(7,1)}	{A,C,G,N,Z}	AT					