4ème IR Systèmes Informatiques NOM : PRENOM :

[i]CONTROLE INTELLIGENCE ARTIFICIELLE (I4RSD11) - 26/03/2024

Durée 1h15. **Seuls documents autorisés : uniquement le support de cours et les notes de TP**
Le sujet peut être utilisé comme partie de la **copie d'examen**.

PREMIERE PARTIE : L'ALGORITHME A* (10 PT)

Exercice 1 : DKT (2 pt)

La ***distance de Kendall-Tau***, notée $D_{KT}(S_1, S_2)$, fournit une mesure pour la distance entre 2 séquences d'objets (permutations) S_1 et S_2.

$D_{KT}(S_1,S_2)$ = **nombre de paires d'objets ordonnées différemment par S_1 et S_2.**

Exemple :

S_1 = [C,D,A,E,B] $D_{KT}(S_1,S_2) = 5$ car parmi les 10 paires possibles, 5 sont mal ordonnées :
S_2 = [A,B,C,D,E] **{(C,A), (C,B), (D,A), (D,B), (E,B)}**.

On définit une opération de **modification** de séquence : choisir **2 objets consécutifs** et les permuter : l'objet en position *i* passe en position *i*+1 et celui en position *i*+1 passe en position *i* avec $1 \leq i < n$:

$U = [O_1, O_2, ..,.\boldsymbol{O_i}, \boldsymbol{O_{i+1}}, ...O_n]$	$\longrightarrow$	$V = [O_1, O_2, ..,.\boldsymbol{O_{i+1}}, \boldsymbol{O_i}, ...O_n]$
Avant	Permuter $\boldsymbol{O_i}$ et $\boldsymbol{O_{i+1}}$	Après

Partant d'une séquence de 5 objets initiale donnée $\boldsymbol{S_0}$ **= [C,D,A,E,B]**) on veut trouver une suite de modifications, de longueur minimale, pour passer de S_0 à ***T*= [A,B,C,D,E]** à l'aide de A*.

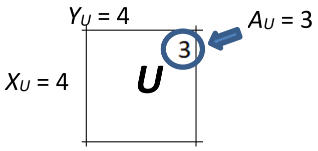
<u>Montrer que ce problème est trivial car $h(U) = D_{KT}(U,T)$ est une heuristique parfaite pour ce problème</u>. (2 pt)

Conseils : montrer d'abord que dans toute situation S différente de T, on peut toujours trouver un échange ***utile*** *(qui fait progresser vers la solution) ; calculer l'évolution des fonction g, h et f lors d'un tel échange. Conclure.*

Exercice 2 : Manhattan en 3D (8 pt)

On désire résoudre avec A* le problème du plus court chemin du point U_0 de coordonnées (1,1) au point T terminal de coordonnées (8,8) dans une grille (voir page2), à l'aide d'actions de déplacements élémentaires (**vers une case immédiatement voisine** : { **b**(as), **d**(roite), **g**(auche), **h**(aut) } envisagées dans cet ordre.

Chaque case *U* de la grille est caractérisée par ses coordonnées cartésiennes (X_U et Y_U) mais aussi par son altitude A_U indiquée dans le coin supérieur droit) ; par exemple la case (4,4) a pour altitude 3 (voir ci-contre).



Aucune case n'est interdite mais on ne peut pas sortir de la grille.

Le coût d'une action de déplacement de la case *u* à une case voisine *v* tient compte à la fois d'un coût de déplacement *horizontal* et d'un coût de déplacement vertical fonction du changement d'altitude (dénivelé) :

$$\boldsymbol{k(u,v) = |X_U - X_V| + |Y_U - Y_V| + |A_U - A_V|} \qquad \forall u, \forall v \in \textit{successeurs(u)}$$

2.1 Fonction *h*

On choisit comme fonction heuristique une fonction composée de la distance de Manhattan classique augmentée de la différence absolue d'altitude entre *U* et *T* :

$\boldsymbol{h(U) = \lvert X_U - X_T\rvert + \lvert Y_U - Y_T\rvert + \lvert A_U - A_T\rvert}$

a. <u>Montrez que $h(U_0) = 16$ (détaillez le calcul)</u> (0,5 pt)

b. <u>Montrez que la fonction *h* est coincidente</u>. (0,5 pt)

c. <u>Montrez que la fonction *h* est monotone</u>. Montrer que la <u>fonction *h* est minorante</u>. (2 pt)

Conseil : vous pouvez utiliser (sans la redémontrer) l'inégalité triangulaire de la valeur absolue :

$$|a-c| \leq |a-b| + |b-c| \qquad \forall a, \forall b, \forall c$$

4ème IR Systèmes Informatiques NOM : PRENOM :

2.3 Développement limité de A* (5 pt)

On ne demande pas de développer intégralement A* et de donner la solution.

<u>Indiquer et placez seulement sur la grille les **12 premiers états** découverts et évalués par A* à partir de</u> U_0.
On prendra bien soin de numéroter les états par ordre croissant U_1, U_2… U_{12} en respectant un ordre lexicographique pour appliquer les actions { **b(**as), **d**(roite), **g**(auche), **h**(aut) }.

	1	2	3	4	5	6	7	8
1	**U0** 0 *f h g* 16 16 0	1	2	2	3	5	5	3
2	2	5	5	3	4	1	1	0
3	3	4	4	3	3	3	3	4
4	5	4	2	3	2	1	1	5
5	2	2	2	3	5	5	5	3
6	4	4	4	3	2	2	2	1
7	5	2	5	5	4	3	1	2
8	6	5	2	0	1	1	2	**T** 2 *f h g* 0

EXERCICE 3 : ALGORITHMES MINMAX ET ALPHA-BETA (6 PT)

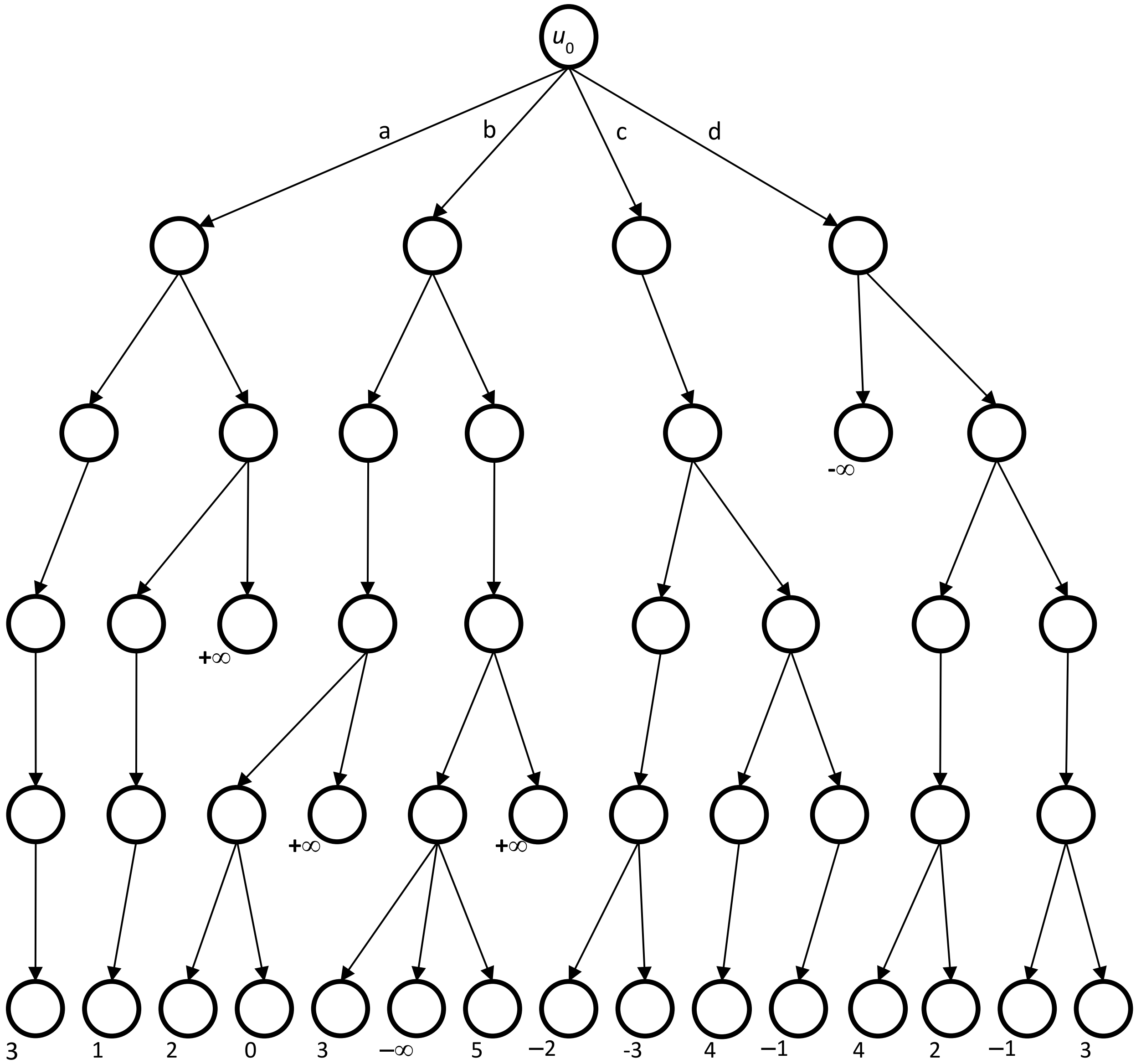
On considère l'arbre de jeu à 2 joueurs développé par l'algorithme ***minmax*** (voir page 4).

Les valeurs de l'heuristique données sur les feuilles sont établies du seul point de vue du 1er joueur (« max »).

3.1 Indiquez la valeur minmax en chaque nœud interne. Quel est le meilleur coup à jouer à la racine ? (2 pt)

3.2 Indiquer (avec un autre couleur ou un e position différente de l'étiquette) la valeur ***negamax***. (1 pt)

3.3 Si on avait appliqué l'algorithme ***Alpha-Beta***, quelles branches de l'arbre de jeu aurait été élaguées ? Placer une croix (×) sur ces branches. (3 pt)



EXERCICE 4 : ALGORITHME AO* (4 PT)

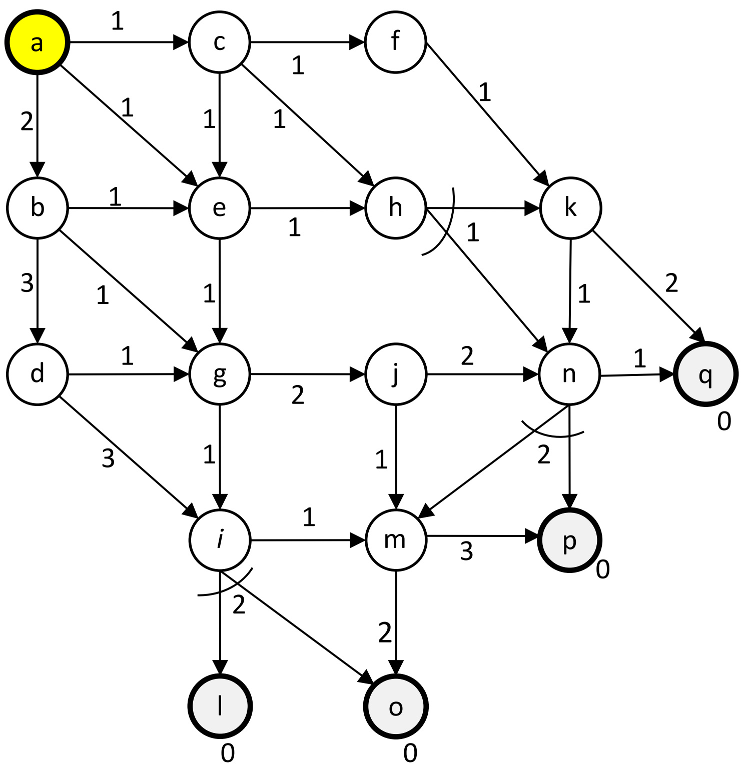
Appliquer l'algorithme AO* à l'hypergraphe ci-dessous, associé à la décomposition du problème **a**.

Dessiner le **sous-graphe solution de côut total minimal** dans le cadre à droite.

Sur ce sous-graphe, indiquez le coût de décomposition de chaque ss-problème de la solution ainsi que celui de a.

Les valeurs de l'heuristique estimant le coût de résolution sont données dans le tableau suivant :

U	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
h(*U*)	5	4	4	4	5	3	3	4	2	2	2	0	2	1	0	0	0



Sous-graphe solution