SOLUTION CONTROLE IA (I4RSD11) - 26/03/2024

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

EXERCICE 1 : DKT (2 PT)

Les 3 premières démonstrations de propositions qui suivent n'étaient pas demandées dans l'énoncé, mais nous les donnons pour combler la curiosité du lecteur.

Proposition

$h(T)$=0. h est coïncidente	(P0)

Preuve

Dans T, aucune paire n'est mal ordonnée. Donc **$h(T)$=0**. □

Proposition

Pour tout état **$U \neq T$, $h(U)>0$**	(P1)

Preuve

Si U n'est pas l'état terminal, il existe au moins une paire d'objets mal ordonnée $\Rightarrow h(U) > 0$. □

Proposition

***h* est monotone**	(P2)

Preuve

$h(U)$ mesure le nombre total n de paires d'objets mal ordonnées dans la séquence U, qu'il s'agisse d'objets *consécutifs* (qui se succèdent strictementnt dans U) *ou pas*. Or on ne peut réaliser une action de permutation qu'entre 2 objets CONSECUTIFS dans U et lorsqu'on permute 2 objets consécutifs, une seule paire mal ordonnée au maximum peut être supprimée. Une action n'est jamais neutre :

$\forall V$ successeur de U, **$h(V) \neq h(U)$** ; ou bien l'action **dégrade la situation** ($h(V) = h(U)$**+1**), ou bien elle l'**améliore** ($h(V) = h(U)$**–1**) ; donc **$h(U)$ - $h(V)$** = $\pm 1 \leq |h(U) - h(V)| = 1 =$ **$k(u,v)$** **h est monotone.** □

Retour à la vraie question posée : on souhaite démontrer que $h(U) = D_{KT}(U,T)$ est une **heuristique parfaite, c-à-d qui estime de façon exacte le coût total optimal pour atteindre l'état but : $h(U_0) = h^*(U_0) = g^*(T)$**.

Proposition

$\forall U \neq T \Rightarrow \exists$ toujours au moins une action utile (améliorante) et optimale à effectuer.	(P3)	(1 PT)

Preuve

On sait que $\forall U$ (**$U \neq T \Leftrightarrow h(U)>0$**) et

$\forall U$ **$h(U)>0$** $\Rightarrow$ il existe au moins une paire d'objets mal ordonnée dans la séquence U.

Comme h comptabilise TOUTES les paires d'objets mal ordonnées : des paires d'objets **consécutifs** et des paires d'objets **non consécutifs** ; il y a 2 cas :

a. une paire d'objets **consécutifs** est mal ordonnée dans U ; alors il existe une action applicable utile évidente et optimale : celle qui consiste à permuter ces 2 objets. CQFD

b. une paire d'objets **non consécutifs** est mal ordonnée dans U ; on ne peut pas échanger directement ces objets. Montrons que dans ce cas, **il existe nécessairement une *autre paire* d'objets consécutifs mal ordonnée** dans U et donc une action utile et optimale qui consiste à permuter cette autre paire.

   Preuve (par l'absurde) Soit $n = |U|$

   Supposons que les (n-1) paires d'objets consécutifs de U : {(U[1],U[2]), (U[2],U[3]), …, (U[n-1],U[n])} soient correctement ordonnées. Alors, par fermeture transitive (la relation d'ordre est une relation transitive), toute paire ($U[i]$, $U[j]$) **$1 \leq i < j \leq n$** est également bien ordonnée, ce qui contredit l'hypothèse initiale selon laquelle il existe une paire d'objets non consécutifs mal ordonnée.

Donc s'il existe une paire d'objets *non consécutifs* mal ordonnée, il existe nécessairement au moins une autre paire d'objets *consécutifs* mal ordonnée, que l'on peut permuter. CQFD

En réunissant les 2 cas a. et b. , on peut conclure que :

$\forall U \neq T \Rightarrow \exists$ **toujours** au moins une paire d'objets **consécutifs** mal ordonnée et donc $\exists$ **toujours une action utile** : **cette action a un coût $k(U,V) = 1$, elle est améliorante** : $h(V) = h(U) - 1$ et **elle est optimale** car on ne peut pas espérer ordonner plus d'une paire à la fois au cours d'une même action. □

Proposition

$\forall U_0 \neq T$ $\exists$ toujours au moins une *suite* d'actions optimales de longueur $h(U_0) = h^*(U_0) = g^*(T)$ (P4) (1PT)

Preuve

D'après P3, $\forall U \neq T$ $\exists$ toujours au moins une action améliorante optimale. Si plusieurs actions améliorantes existent à une étape donnée, elles sont toutes optimales, et il suffit de choisir la "première" qui se présente, par exemple celle qui contient l'objet situé le plus à gauche dans *T* ; par la suite l'heuristique permettra de ne jamais remettre en cause cette permutation dans le future (sinon *h* augmente) : toute paire une fois ordonnée correctement le reste définitivement.

Par définition, entre 2 états successifs *U* et *V*, on a : $k(U, V) = 1$, donc $\boldsymbol{g(V) = g(U) + 1}$

D'autre part, la modification effectuée est toujours améliorante : $\boldsymbol{h(V) = h(U) - 1}$

Globalement : $\boldsymbol{f(V)} = g(V) + h(V) = g(U)+1 + h(U) - 1 = g(U) + h(U) = \boldsymbol{f(U)}$ ***f* est constante**

Donc $\forall U_0 \neq T$, il existe une suite d'actions optimales de U_0 jusqu'à *T* : il suffit à chaque étape de permuter la "première" paire d'objets consécutifs mal ordonnée qui se présente ; l'algo est glouton.

$f(U_0) = f(U_1) = f(U_2) = f(U_3) = \ldots = f(T)$. Or $\boldsymbol{g(T) = g^*(T)}$ car chaque action appliquée est optimale. De plus $f(U_0) = h(U_0)$ car $g(U_0)=0$, et

$f(T) = g(T)$ car $h(T)=0$, donc $\boldsymbol{h(U_0)} = f(U_0) = f(T) = \boldsymbol{g^*(T) = h^*(U_0)}$

***h*(*U*) est une heuristique parfaite**. Elle mesure de façon exacte le nombre d'actions à réaliser jusqu'à *T*.

□

EXERCICE 2 DEPLACEMENTS EN 3D (8 PT)

2.1 Fonction *h*

L'heuristique choisie est **la somme de l'heuristique de Manhattan classique en 2D et d'une heuristique basée sur la différence d'altitude en valeur absolue** entre *U et T*. C'est une **distance de Manhattan en 3D**.

$$h(U) = d_{Manhattan}(U,T) + h_{alt}(U) = |X_U-X_T|+|Y_U-Y_T| + |A_U-A_T|$$

a. **Calcul de $h(U_0)$**

On a : $|X_U - X_T|=7$, $|Y_U - Y_T|=7$ et $|A_U - A_T|=2$ d'où $h(U_0)= 7+7+2=\mathbf{16}$ □ (1 pt)

b. ***h* est-elle coincidente ?**

Trivial : si *U*=*T*, les 3 coordonnées sont identiques et $h(U) = h(T) = 0$ (0,5 pt)

c. ***h* est-elle monotone ?**

On réutilise -sans la démontrer- l'inégalité triangulaire de la fonction valeur absolue :

$$\forall a,\forall b,\forall c,\ |a-c| \leq |a-b| + |b-c| \quad \text{(IT1)}$$

qu'on peut écrire également (en faisant passer $|b-c|$ à gauche de l'inégalité :

$$\forall a,\forall b,\forall c,\ |a-c| - |b-c| \leq |a-b| \quad \text{(IT2)}$$

On développe $\underline{h}(U) - \underline{h}(V)$ $= |X_U-X_T| + |Y_U-Y_T| + |A_U-A_T| - (|X_V-X_T| + |Y_V-Y_T| + |A_V-A_T|)$

On regroupe chaque paire de termes relatifs à la même coordonnée (*X*, *Y* et *A*) :

$$h(U) - h(V) = \underbrace{(|X_U-X_T|-|X_V-X_T|)}_{P1} + \underbrace{(|Y_U-Y_T| - |Y_V-Y_T|)}_{P2} + \underbrace{(|A_U-A_T| - |A_V-A_T|)}_{P3}$$

On applique l'inégalité triangulaire (IT2) à chaque paire de termes P_i i=1,2,3

$$(|X_U - X_T| - |X_V - X_T|) \leq |X_U - X_V|$$
$$(|Y_U - Y_T| - |Y_V - Y_T|) \leq |Y_U - Y_V|$$
$$(|A_U - A_T| - |A_V - A_T|) \leq |A_U - A_V|$$

En **sommant** les 3 inégalités qui précèdent, on obtient une nouvelle inégalité:

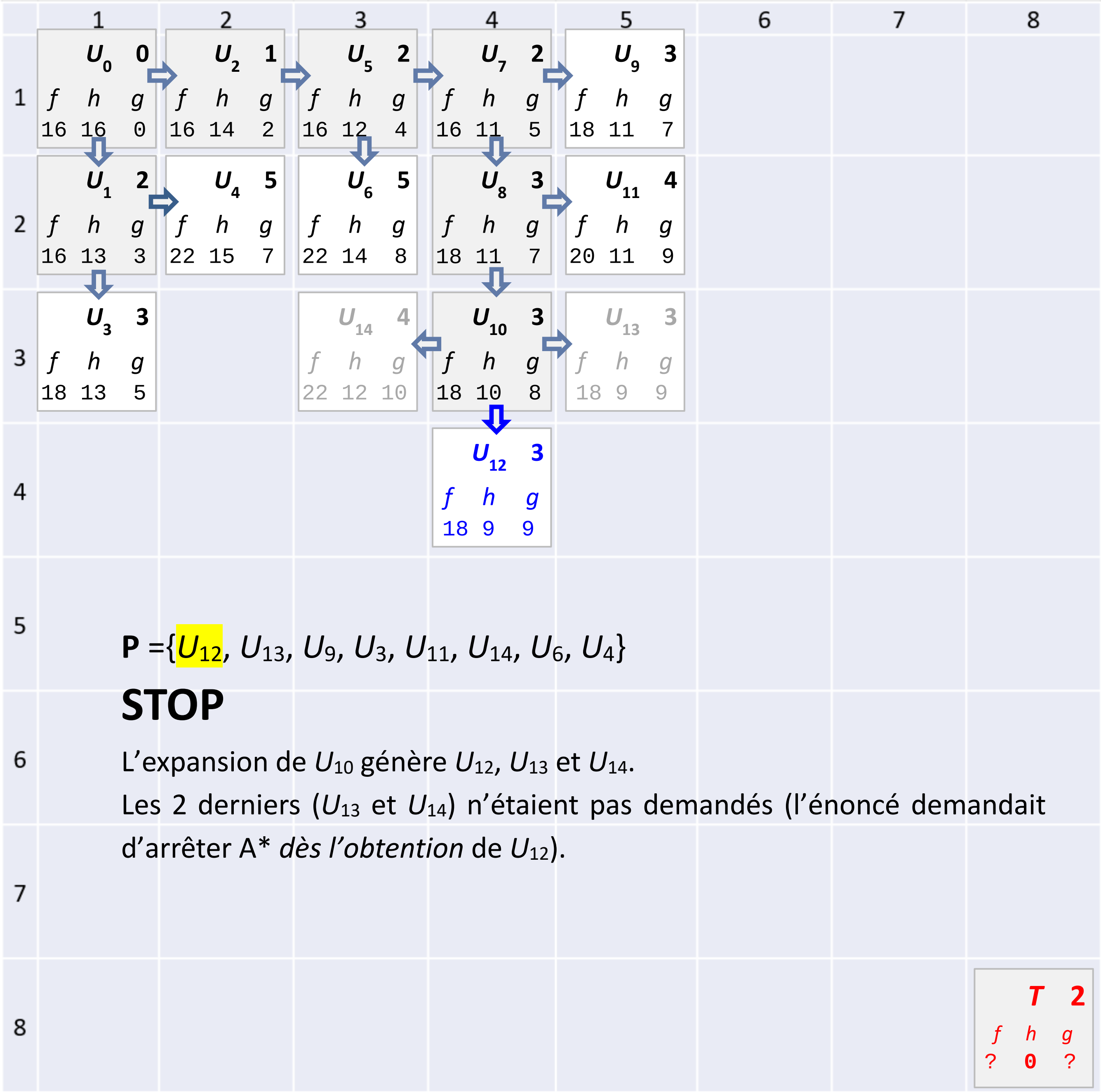
$$h(U) - h(V) = (|X_U - X_T| - |X_V - X_T|) + (|Y_U - Y_T| - |Y_V - Y_T|) + (|A_U - A_T| - |A_V - A_T|) \leq |X_U - X_V| + |Y_U - Y_V| + |A_U - A_V|$$

or le membre de droite $|X_U - X_V| + |Y_U - Y_V| + |A_U - A_V| = k(U,V)$, d'où :

$h(U) - h(V) \leq k(U,V)$ $\Leftrightarrow$ ***h* est monotone** CQFD □ (2 pt)

***h* est monotone et coïncidente** $\Leftrightarrow$ ***h* est minorante** CQFD □ (0,5 pt)

2.2 Développement de A* jusqu'à génération du l'état U_{12} : (4 pt)



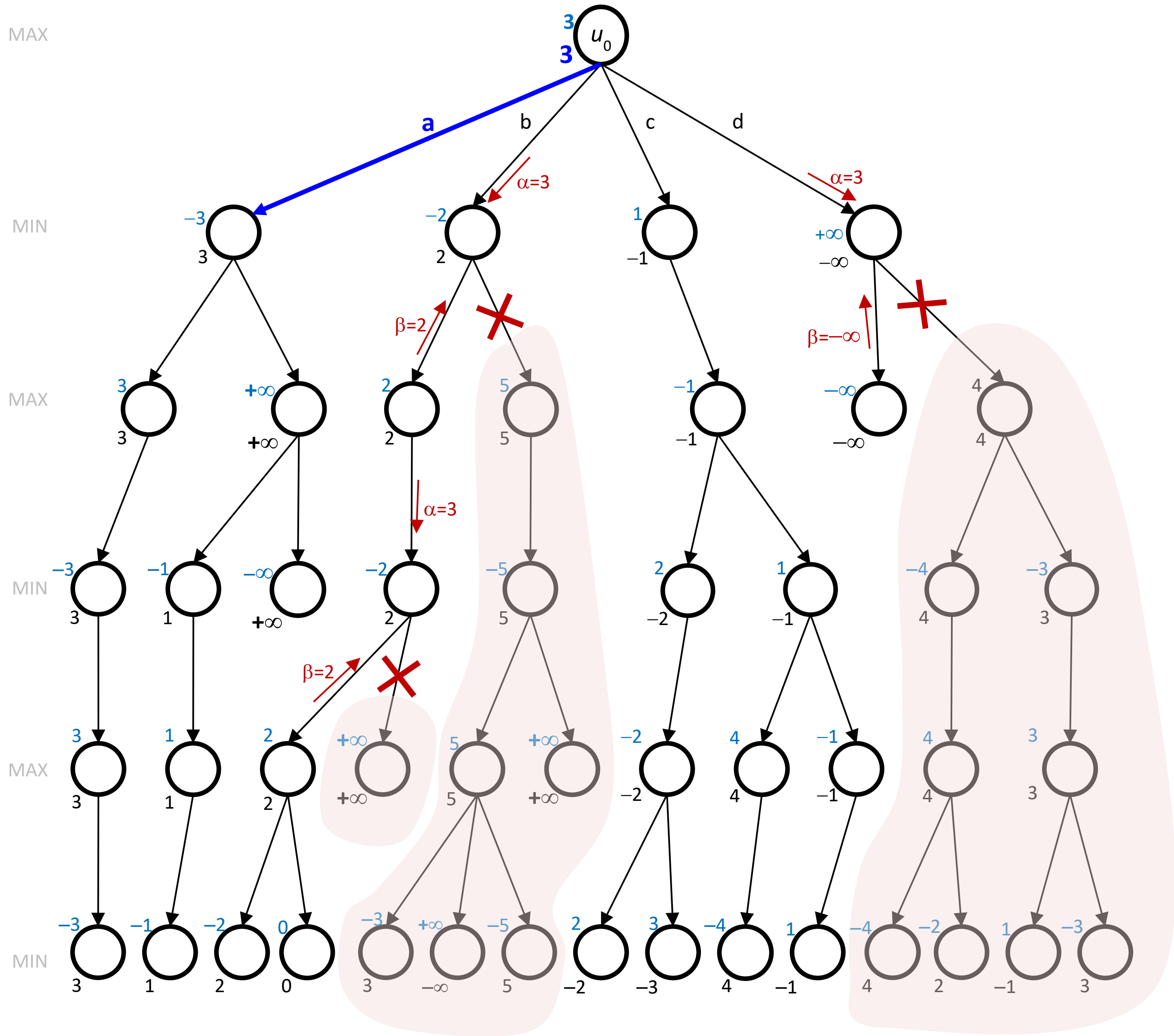
P ={U_{12}, U_{13}, U_9, U_3, U_{11}, U_{14}, U_6, U_4}

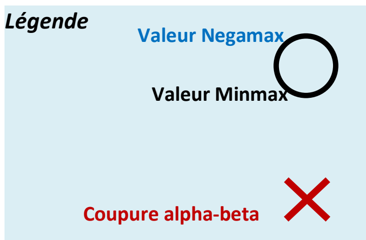
STOP

L'expansion de U_{10} génère U_{12}, U_{13} et U_{14}.

Les 2 derniers (U_{13} et U_{14}) n'étaient pas demandés (l'énoncé demandait d'arrêter A* *dès l'obtention* de U_{12}).

EXERCICE 3 : ALGORITHMES MINMAX ET ALPHA-BETA (5 PT)



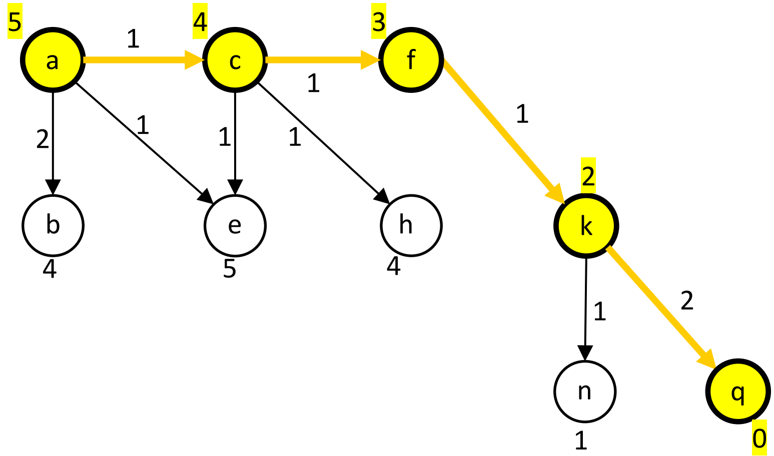


Valeurs minmax correctes et solution trouvée (coup *a*) (1,5 pt)

Valeurs negamax correctes (0,5 pt)

Coupures alpha-beta correctes (2 pt)

EXERCICE 3 : ALGORITHME AO* (5 PT)



Sous-graphe solution (en jaune) de coût ***q*(a)= 5**
Les nœuds a puis c puis f puis k ont été développés.
q étant résolu, tous ses ascendants le sont aussi.

Les valeurs des noeuds résolus (a, c, f, k, q) sont les coûts réels définitifs

Les valeurs des autres noeuds (b, e, h, n) sont les valeurs de l'heuristique. Ces noeuds sont atteints par AO* mais ils ne sont pas pas intéressants car l'heuristique étant minorante, ils ne peuvent pas améliorer le coût de leurs sommets ascendants $\Rightarrow$ ils ne sont donc pas développés.

Dans ce problème, la fonction heuristique est particulièrement efficace puisque l'algorithme ne procède à aucune révision des coûts et ne fait donc aucun retour arrière.

Sous-graphe ok (2,5 pt)

Valeurs coûts exactes (2,5 pt)