Examen Final
Juin 2016

Coefficient : 40%.
Aucun document autorisé.

Remarques et conseils :

1) Lisez **attentivement** chaque question avant d'y répondre.
2) Indiquez clairement sur votre copie le numéro de l'exercice avant d'y répondre.
3) Lorsque vous définissez un prédicat, son profil et sa définition formelle doivent être au moins indiqués. Un jeu d'essais est facultatif.
4) Expliquez autant que possible vos choix lors de la définition d'un prédicat ou d'une fonction.
5) Le barème défini ci-après est susceptible d'être modifié.

Exercice 1 : Questions diverses (5 points)

1. Soit le CSP (X,D,C) suivant :
   - X = {X1; X2; X3; X4} l'ensemble des variables,
   - D = {D1,D2,D3,D4} leurs domaines de valeurs, avec D1=D2=D3=D4={1,2,3,4,5},
   - C = {(C1,R1),(C2,R2),(C3,R3),(C4,R4)} l'ensemble des contraintes, telles que C1 = (X1, X4), C2 = (X1, X2), C3 = (X1, X3), C4 = (X4,X2), R1 = {(1,3),(1,2),(2,5),(3,2)}, R2={(1,3),(2,2),(3,3)}, R3={(1,1),(2,2),(3,3)}, R4={(1,2),(2,3),(5,3)}.

   Représentez ce CSP sous la forme d'un graphe et donnez toutes les solutions possibles. **(3 points)**
2. Quel type de problème permet de résoudre l'algorithme *Iterative Deepening* ? En quoi consiste cet algorithme ? **(1 point)**
3. Définissez aussi précisément que possible ce qu'est un problème de satisfaction de contraintes. Donnez plusieurs exemples. Rappelez la terminologie pour ce type de problème. **(1 point)**

Exercice 2 : Prolog (5 points)

1. Définissez le prédicat *groupe(L,G)* qui vérifie si la liste des occurrences G comporte les nombres corrects d'occurrences des éléments apparaissant dans L.
   Par exemple, le prédicat *groupe([a,c,a,b,a],[[a,3],[c,1],[b,1]])* renvoie *yes* car il y a 3 occurrences de a, 1 occurrence de b et 1 occurrence de c dans la liste *[a,c,a,b,a]*.
   Pour faciliter l'écriture de ce prédicat, il est recommandé de définir et d'utiliser le prédicat *occurrence(E,L,N)* qui est vrai si N est le nombre d'occurrences de E dans L. **(3 points)**
2. Définissez le prédicat *sup_k(L,R,N)* qui est vrai si la liste R est obtenue en supprimant le N^{ème} élément de L. Par exemple, *sup_k([a,b,c,d],R,2)* renvoie R=[a,c,d]. **(2 points)**

(Suite page suivante)

Exercice 3 : Algorithme Alpha-Bétâ (5 points)

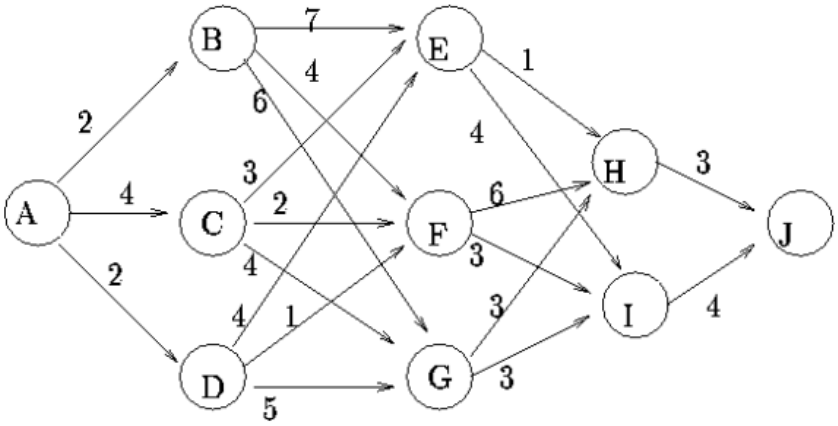
Nous voudrions utiliser l'algorithme Alpha-Bétâ pour savoir qui est le vainqueur du jeu de Nim suivant.
On part d'une pile de n jetons. A tour de rôle, un des deux joueurs doit diviser en deux piles non vides et de tailles différentes une des piles devant lui. A chaque tour de jeu une nouvelle pile est donc créée. Par exemple, à partir d'une configuration de jeu comportant 3 piles respectivement composées de 2, 3 et 2 jetons, le seul coup jouable est de diviser la seconde pile en 2 et 1 jeton. Le joueur qui ne peut plus jouer a perdu.
L'IA sera le premier à jouer et elle est supposée maximiser ses gains, tandis que l'adversaire les minimise.

Questions :

1. Appliquez l'algorithme Alpha-Bétâ en considérant une pile de départ de **n=9** jetons. Précisez pour chaque nœud de l'arbre de jeu ses bornes finales [alpha; bétâ]. Indiquez également avec une croix les coups qui seront élagués par l'algorithme et donnez enfin le premier coup qu'effectuera l'IA.
2. Qui gagne ?

Exercice 4 : Algorithme A* (5 points)

Soit le graphe suivant où est indiqué sur chaque arc le coût de passage d'un nœud vers un autre.



On a de plus la fonction heuristique h donnée par le tableau suivant, qui estime le coût pour atteindre le nœud J à partir de n'importe quel autre nœud.

	A	B	C	D	E	F	G	H	I	J
h	16	14	10	11	13	7	4	7	6	0

Question : Appliquez l'algorithme A* pour déterminer le plus court chemin entre A et J en utilisant la fonction h donnée précédemment.
Vous représenterez le déroulement de l'algorithme A* sous forme de tableau. Chaque ligne du tableau correspond à une itération de l'algorithme. La première colonne indique le numéro d'itération, la deuxième le nœud choisi, la troisième l'ensemble des nœuds ouverts (nœuds contenus dans l'*Open List*) et la quatrième colonne les nœuds fermés (nœuds contenus dans la *Closed List*). Chaque état est de la forme sommet(sommet père,g(x),f(x)=g(x)+h(x)). Ci-dessous le début du tableau à compléter.

Itération	Noeud choisi	Noeuds ouverts	Noeuds fermés
0		{A(-,0,16)}	{}
1	A(-,0,16)	...	...