Examen Final
Mercredi 19 Janvier 2011, de 8h à 10h

Coefficient : 40%.
Aucun document autorisé.

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

Partie I (10 points)

Exercice 1 : Fermier & Co. (7 points)

Un fermier doit traverser une rivière avec une salade, un lapin et un chien. Il dispose d'une embarcation qui lui permet de ne prendre qu'un objet ou un animal avec lui.
Nous souhaitons déterminer la procédure qu'il doit mettre en œuvre pour réaliser sa traversée.
Attention, les animaux sont affamés. Si le fermier n'est pas présent, le lapin fera de la salade sont déjeuner. Mais cet affamé risque de terminer dans le ventre du chien si le fermier n'est pas là pour calmer la fringale de cet animal.

Questions :
1. Donnez une représentation d'un état de ce système. (1 point)
2. Donnez un chemin possible de résolution à partir de l'état initial où tout le monde est à gauche de la rivière. (1 point)
3. Définissez les règles de transition entre états. (2 points)
4. Ecrivez le prédicat ***regle(+E1, ?E2)*** qui permet de passer d'un état à un autre en appliquant les opérations définies. Si possible, limitez autant que possible le nombre de règles du prédicat (1 point).
5. Ecrivez un prédicat ***chemin(?Ch)*** permettant de connaître la succession des étapes par lesquelles doit passer le fermier pour réaliser sa traversée de gauche à droite sans perte irrémédiable (2 points).
   chemin(L) donnera L=[*Ei, E2, E3, ... , En, Ef*] où *E2, ... , En* sont les états intermédiaires entre l'état initial *Ei* et l'état final *Ef*.

Exercice 2 : Neurone artificiel en Prolog et Lisp (3 points)

Considérons que l'équation d'activation d'un neurone artificiel dans un perceptron (un type de réseau de neurones artificiels) peut être représentée par :

$$f(x,w)=\sigma\left(\sum_{i=0}^{N} x_i w_i\right) \text{ avec } \sigma(v)=\begin{cases}1 \text{ si } v>0\\ 0 \text{ sinon}\end{cases}, x\in\mathbb{R}^N,\ w\in\mathbb{R}^N \text{ et } v\in\mathbb{R}$$

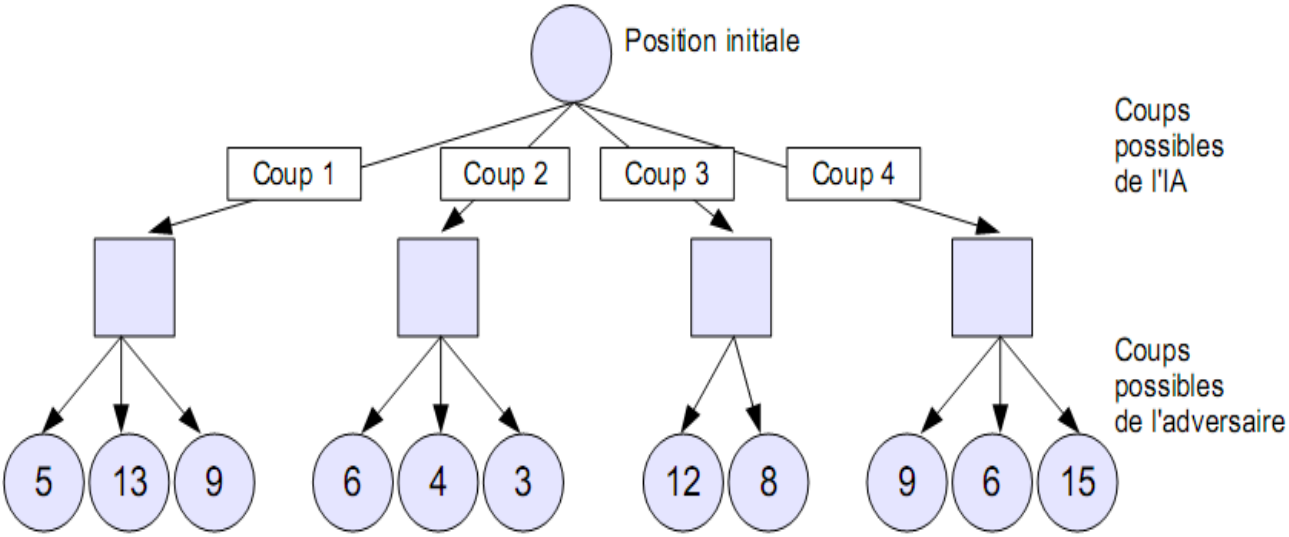
x représente le vecteur d'entrées du neurone, w sont les poids associés à chacune des entrées. Les vecteurs x et w de N+1 composantes peuvent être représentés par des listes à N+1 éléments.

Questions :
1. Ecrivez le prédicat Prolog implémentant cette équation d'activation. (1,5 points)
2. Ecrivez la fonction Lisp implémentant cette équation d'activation. (1,5 points)

Partie II (10 points)

Exercice 3 : Algorithme Minimax et élagage Alpha-Bétâ (5 points)

L'arbre ci-dessous représente l'arbre de recherche d'un jeu à deux joueurs. L'évaluation de chaque position pouvant être atteinte à un niveau de profondeur 2 à partir de la position initiale est indiquée sur les feuilles de l'arbre.

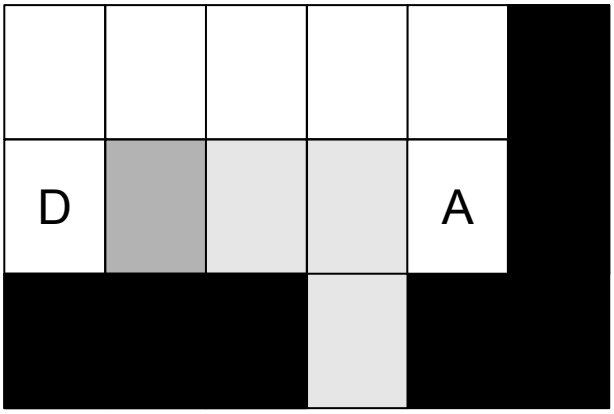


Questions :

1. Précisez quel est l'objectif de l'I.A. et quel est l'objectif de son adversaire.
2. En utilisant l'algorithme Minimax, précisez pour chaque noeud son évaluation et indiquez le coup qu'effectuera l'I.A.
3. En utilisant l'élagage Alpha-Bétâ, précisez pour chaque noeud ses bornes finales [Alpha; Bétâ]. Indiquez également avec une croix les coups qui sont élagués par l'algorithme et donnez enfin le coup qu'effectuera l'I.A.
4. Quel critère dans la construction de l'arbre de recherche permettrait d'améliorer l'efficacité de l'élagage Alpha-Bétâ ? Si ce critère est appliqué, indiquez les noeuds dans l'arbre de recherche ci-dessus qui seraient alors élagués.

Exercice 4 : Algorithme A* (5 points)

La figure ci-dessous représente l'environnement d'un personnage virtuel, que nous nommerons Jack, qui doit se déplacer de la case D (départ) vers la case A (arrivée).



A chacun de ses déplacements cardinaux (haut, bas, droite ou gauche), Jack consomme des points d'énergie, si bien qu'il désire atteindre A à partir de D en empruntant le chemin le moins coûteux en énergie.
Les cases noires représentent des obstacles infranchissables : aucune de celles-ci ne peuvent être atteintes par Jack.

Un point d'énergie est dépensé lorsque Jack arrive sur une case blanche, 2 points lorsqu'il arrive sur une case grise claire et 3 points sur une case grise foncée.
Pour résoudre le problème dont doit faire face Jack, utilisons l'algorithme A*...

Questions :

1. Donnez une représentation d'un état pour résoudre le problème exposé ci-dessus avec A*.
2. Exprimez la fonction successeur **succ(x)** pour chacun des états **x** nécessaires à la résolution.
3. Est-ce que la distance de Manhattan est une heuristique admissible ? Pourquoi ? Etayez votre réponse en donnant un exemple.
4. Proposez une heuristique, si possible admissible, et résolvez ce problème avec l'algorithme A* en indiquant à chaque itération de l'algorithme :
   - le contenu de la liste Open,
   - le contenu de la liste Closed,
   - les valeurs de g, de h et de f pour le noeud courant extrait de la liste Open.

   On suppose qu'à l'itération 1, la liste *Open* ne contient que le noeud correspondant à l'état initial D.

Itération	**Liste *Open***	**Liste *Closed***	**Noeud N extrait de la liste *Open***	**Valeur g du noeud N**	**Valeur h du noeud N**	**Valeur f du noeud N**
1						
2						
3						
4						
...						

5. Combien de noeuds sont développés sans utiliser d'heuristique (c'est-à-dire que h(x) = 0 quel que soit l'état x)

Rappel : la distance de Manhattan est définie par : $d((x_1,y_1), (x_2,y_2)) = |x_1-x_2| + |y_1-y_2|$.