UNIVERSITÉ DE SHERBROOKE
Département d’informatique

IFT 615
Intelligence artificielle

Examen périodique
Hiver 2010

Le samedi 20 février, 9 h 00 à 10 h 50, au D3-2038

PROFESSEUR
Froduald Kabanza
planiart.usherbrooke.ca/kabanza

SOLUTIONS

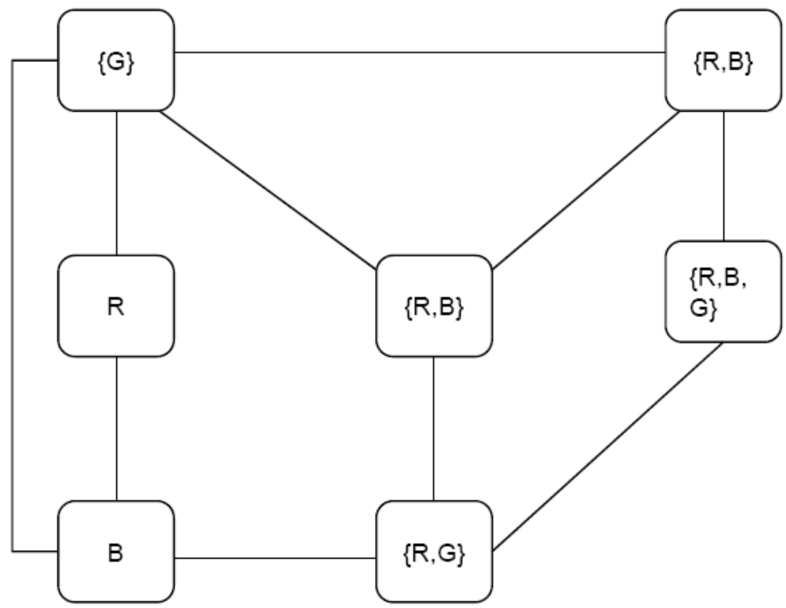
Question 1 (2 points) – Algorithme A*

Pour l'algorithme A*, si h_1 et h_2 sont deux fonctions heuristiques admissibles, lesquelles des fonctions suivantes sont aussi admissibles. Encerclez les bonnes réponses.

1. $h_1 + h_2$
2. $h_1 * h_2$
3. ***$max(h_1,h_2)$***
4. ***$min(h_1,h_2)$***
5. ***$(\alpha)h_1 + (1-\alpha)h_2$, pour $0 \leq \alpha \leq 1$***
6. ***$(\alpha)h_1$, pour $0 \leq \alpha \leq 1$***
7. ***$(1-\alpha)h_2$, pour $0 \leq \alpha \leq 1$***

Question 2 (4 points) – Satisfaction des contraintes

a. (3 points) Dans le graphe suivant, chaque nœud représente une variable d'un problème CSP. Un arc entre deux nœuds modélise la contrainte que les variables correspondantes ne peuvent avoir la même valeur. Le domaine de chacune des variables est l'ensemble {R,G,B}. Supposons qu'on est en train d'exécuter l'algorithme *backtracking-search* avec AC-3 (*arc-consistency*-3) et que nous sommes rendus à une profondeur de niveau 2, ayant assigné des valeurs à deux variables, respectivement R et B, tel qu'indiqué sur le graphe. Montrez le résultat de AC-3 à cette étape en inscrivant les valeurs pour chaque variable à l'intérieur du nœud correspondant.

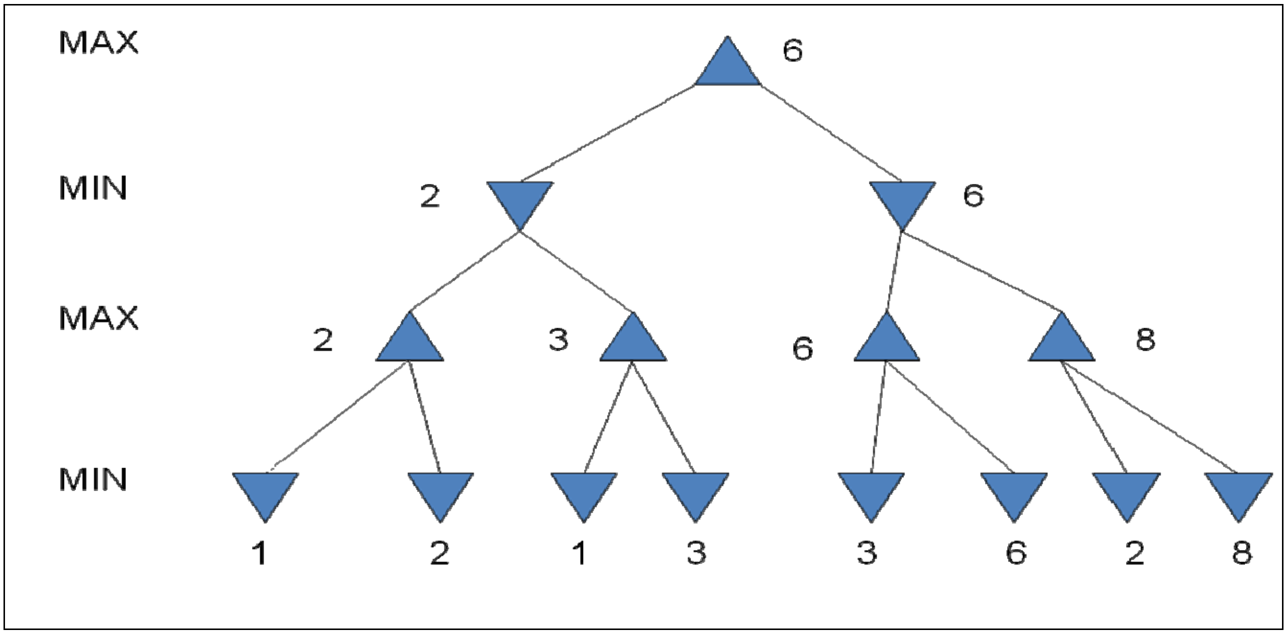


b. (1 point) L'algorithme *backtracking-search* avec AC-3 trouvera-t-il une solution à ce problème? Expliquez votre réponse.

Oui, ce problème a une solution et *backtracking-search* avec AC-3 va la trouver. En fait, il la trouverait à la prochaine étape en assignant R au nœud en bas à droite.

Question 3 (4 points) – *Alpha-Beta Pruning*

a. (2 points) Soit l'espace d'états suivant modélisant les actions de deux joueurs (MAX et MIN). Les feuilles correspondent aux états terminaux du jeu. Les valeurs des états terminaux sont indiquées en bas de chaque état. Indiquez juste à côté de la racine la valeur correspondante selon l'algorithme *alpha-beta pruning* et encerclez les nœuds qui ne seraient pas explorés par ce dernier, en supposant qu'il explore l'espace d'états de la gauche vers la droite.



b. (1 point) Considérons deux fonctions d'évaluation heuristiques de la valeur (utilité) d'un nœud, *h1(n)* et *h2(n)=h1(n)+c*, avec *c* une constante quelconque. Si *alpha-beta pruning* utilise l'une ou l'autre de ces fonctions pour couper la profondeur de recherche va-t-il oui ou non explorer le même espace de recherche? Justifiez votre réponse.

Oui, *alpha-beta pruning* explorera le même espace de recherche avec l'une ou l'autre de ces fonctions. En effet, h2(n) préserve la grandeur relative des nœuds.

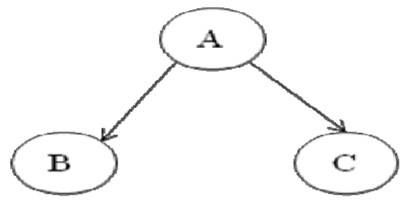
c. (1 point) Si *alpha-beta pruning* utilise une fonction heuristique d'évaluation de la valeur d'un nœud afin de couper la profondeur de recherche garantit-il de calculer un coup optimal pour MAX? Justifiez votre réponse.

Non, *alpha-beta pruning* avec la coupe des nœuds ne garantit pas de trouver une solution optimale. La seule façon de garantir l'optimalité est d'explorer jusqu'aux feuilles [On pourrait aussi tenter de définir une notion d'admissibilité]

Question 4 (4 points) – Réseaux bayésiens

Soit le réseau bayésiens (RB) suivant pour trois variables booléennes aléatoires (A, B et C) avec les tables de probabilité conditionnelle correspondantes.

P(A=T)
0.1



A	P(B=T|A)
T	0.7
F	0.2

A	P(C=T|A)
T	0.4
F	0.6

a. (1 point) Calculez P(B = F, C = T | A=T).

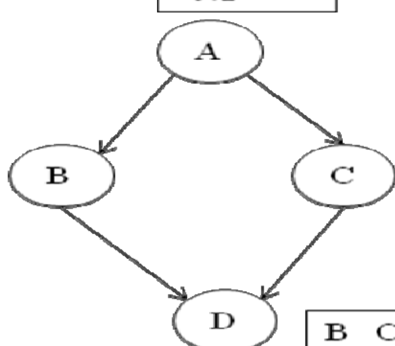
0.12
Vu que B et C sont indépendants étant donné A, P(B,C|A) = P(B|A).P(C|A). Ainsi :
P(B = F, C = T | A=T) = P(B = F | A=T) P(C = T | A=T) = 0.3*0.4=0.12
Preuve que si B et C sont indépendants alors la distribution P(B,C|A) est égale à P(B|C).P(C|A).
P(B,C|A) = P(B,C,A)/(PA)= P(A).P(B|A).P(C|A)/P(A)=P(B|A).P(C|A).

b. (1 point) Calculez P(A=T | B = F, C = T).

=0.012/0.444 = 1/37 = 0.027027027
Selon la règle de Bayes : P(X|Y)=P(Y|X)*P(X)/P(Y). Ainsi :
P(A=T | B = F, C = T)=P(B=F,C=T|A=T)*P(A=T) / P(B=F,C=T)=(0.12*0.1)/P(B=F,C=T)
En faisant la sommation sur la variable caché (A) : P(B=F,C=T) = P(B=F,C=T,A=T)+P(B=F,C=T,A=F).
= P(B=F|A)*P(C=T|A=T)*P(A=T) + P(B=F|A=F)*P(C=T|A=F)*P(A=F)
= (0.3*0.4*01) + (0.8*06*0.9)=0.444

Supposons qu'on ajoute une quatrième variable booléenne, D, selon le graphe suivant.

P(A=T)
0.1



A	P(B=T|A)
T	0.7
F	0.2

A	P(C=T|A)
T	0.4
F	0.6

B	C	P(D=T|B,C)
T	T	0.8
T	F	0.6
F	T	0.5
F	F	0.1

c. (1 point) Pour ce nouveau réseau, est-ce vrai ou faux que A est conditionnellement indépendant de D étant donné B? Justifiez brièvement votre réponse.

Faux. A n'est pas conditionnellement indépendant de D étant donné C, puisque A et D peuvent être corrélés/dépendants à travers B.

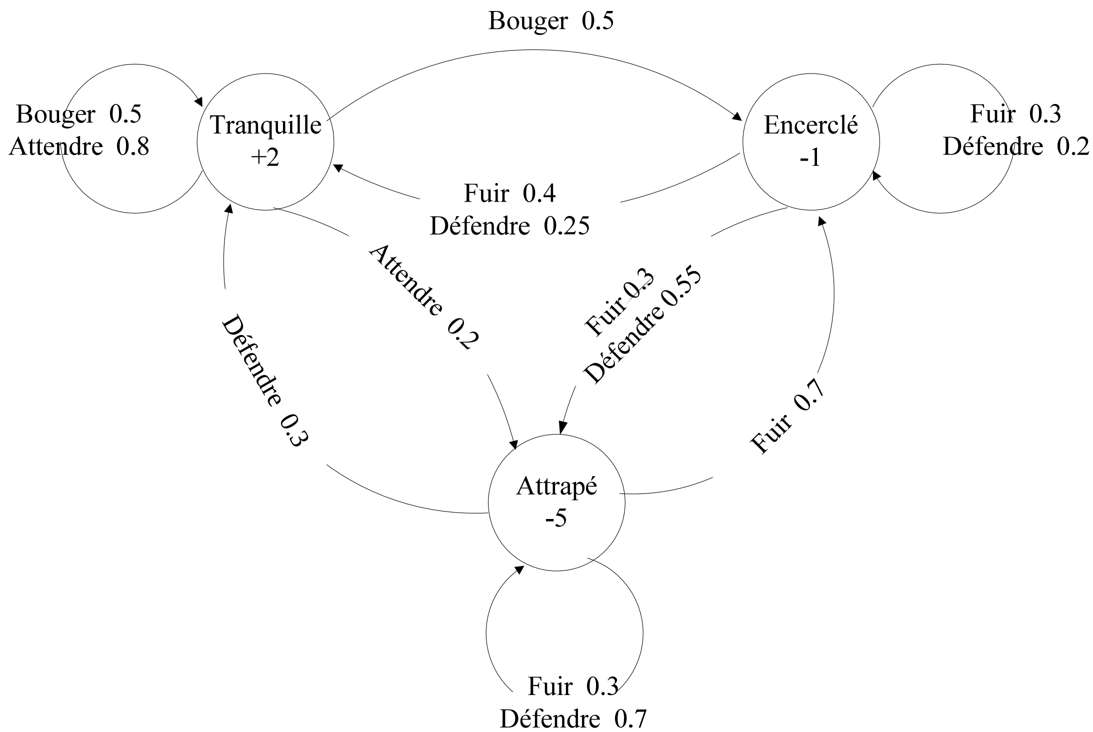
En fait, A est conditionnellement indépendant de D étant donnés B et C.

d. (1 point) Toujours pour ce nouveau réseau, est-ce vrai ou faux que B est conditionnellement indépendant de C étant donné A? Justifiez brièvement votre réponse.

Oui. B est conditionnellement indépendant de C étant donné A. En général, un nœud est conditionnellement indépendant de ses non descendent étant donnés ses parents.

Question 5 (4 points) Processus de décision de Markov

Soit le processus de décision markovien suivant, dont les transitions sont étiquetées par les noms des actions et les probabilités de transitions; les états sont étiquetés par les récompenses correspondantes. Il modélise les actions possibles d'un personnage pourchassé par d'autres. Dans l'état Tranquille, il ne se passe rien – aucun ennemi à l'horizon. Dans cet état il reçoit une récompense de +2. S'il attend sans rien faire, il risque d'être localisé et attrapé par l'ennemi (probabilité de 0.2). S'il est attrapé il a une pénalité de -5. Dans l'état Tranquille, s'il bouge, il a 50% de chances de tomber en embuscade et d'être encerclé par l'ennemi. Une fois encerclé ou attrapé, il peut tenter de fuir ou de se défendre.



a. (3 points) Dans le tableau suivant, indiquez les valeurs d'utilité des états à la fin de chacune des deux premières itérations de l'algorithme *value-itération*, en supposant qu'on utilise un facteur d'atténuation (*discount factor*) de 0.9 et en partant initialement (itération 0) avec des valeurs des états toutes égales à 0. Indiquez seulement les valeurs, ne détaillez pas les calculs.

	Valeurs après la première itération 1	**Valeurs après la deuxième itération**
Tranquille	**2**	**2.54** max(2 + .9*((.5 *2) + (.5 * -1)), 2 + .9*((.8 *2) + (.2 *-5)) max(2.45,2.54)
Encerclé	**-1**	**-1.9** max(-1 + .9*((.4 *2) + (.3 *-1) + (.3 *-5)), -1 + .9*((.25 *2) + (.2 *-1) + (.55 *-5))) Max(-1.9,-3.205)
Attrapé	**-5**	**-6.98** max(-5 + .9*((.7 *-1) + (.3 *-5)), -5 + .9*((.3 *2) + (.7 *-5)) max(-6.98,-7.61)

b. (1 point) Donnez le meilleur plan d'actions (*policy*) résultant de l'itération 2.

Tranquille : Attendre

Encercle : Fuir

Attrape : Fuir

Question 6 (2 points) – Planification de trajectoires

a. (1 point) Lorsqu'on calcule une trajectoire pour un corps articulé en présence de contraintes différentielles, pourquoi n'est-il plus possible d'utiliser directement les approches exactes ou approximatives propres aux problèmes dépourvus de telles contraintes?

Les approches pour les problèmes dépourvues de contraintes différentielles ont comme hypothèse qu'il y a un planificateur local pouvant connecter deux états ou jalons (*milestones* en anglais). Plus spécifiquement, le planificateur local vérifie l'absence de collisions sur un segment reliant deux états. Un tel planificateur local devient difficile à concevoir lorsqu'on a des contraintes différentielles. Par exemple, un véhicule ne peut pas se déplacer latéralement, et une trajectoire en ligne droite n'est pas possible dans les cas où il doit effectuer ce genre de manœuvre (latérale).

b. (1 point) Quelle est la différence entre « complétude » (*completeness* en anglais) et « complétude probabiliste » (*probabilistic completeness* en anglais) pour un algorithme de calcul de trajectoires ? Nommez une approche algorithmique pour chacune de ces deux propriétés.

Un algorithme est complet s'il garantit de trouver une solution si elle existe, et rapporte l'absence de solution si aucune solution n'existe.

La *complétude probabiliste* est plus faible que la complétude tout court, parce qu'elle elle garantit la découverte d'une solution (si elle existe) avec une probabilité convergeant vers 1 lorsque le temps de calcul converge vers l'infini; elle mais ne peut garantir la découverte d'une solution en un temps fini.