UNIVERSITÉ DE SHERBROOKE
Département d'informatique

IFT 615
Intelligence artificielle

Examen périodique
Hiver 2009

Le vendredi 27 février, 10 h 30 à 12 h 20, au D3-2037

PROFESSEUR
Froduald Kabanza
http://www.planiart.usherbrooke.ca/kabanza

SOLUTIONS

Question 1 (5 points) – Satisfaction des contraintes

On vous demande de concevoir un programme pour assigner des sièges lors d'une réception d'un mariage. Nous avons N invités et M tables circulaires, chacune avec une capacité de C (nombres de personnes maximales par table). On suppose que le nombre de tables est suffisant pour le nombre d'invités ($MC \geq N$). Répondez le plus clairement possible aux questions suivantes dans l'espace indiqué.

a. (1 point) Supposons qu'on a des contraintes sur les personnes qui doivent s'asseoir aux mêmes tables ou aux tables adjacentes (par exemple des parents avec leurs enfants) et des paires de personnes qui ne peuvent s'asseoir aux mêmes tables ou à des tables adjacents (par exemple les ex-conjoints). Expliquez comment vous formaliseriez ce problème comme un problème de satisfaction de contraintes.

- *Définir une variable pour chaque invité : N variables au total.*
- *Le domaine de chaque variable est l'ensemble des places : MN variables.*

  (On pourrait utiliser un seul identifiant pour chaque siège ou deux (table,place-sur-la-table)).
- *On définit une fonction d'appartenance aux mêmes tables pour deux places données.*
- *On définit une fonction d'appartenance à deux tables adjacentes pour deux places places données (ou d'adjacence de tables si on a choisi la deuxième représentation) en fonction de la disposition des tables.*
- *Pour chaque groupe de personnes devant s'asseoir aux mêmes tables ou aux tables adjacentes, on définit des contraintes correspondantes.*
- *Même choses pour les personnes qui ne doivent pas s'asseoir au mêmes tables ou aux tables adjacentes*

b. (1 point) Proposez une technique (algorithme) pour résoudre ce problème tel que formulé dans votre réponse précédente.

Backtracking search avec MRV et AC3.

c. (1 point) Tel qu'énoncé au point (a), les contraintes sont « dures » (elles doivent être absolument satisfaites) et le problème pourrait ne pas avoir de solution. Proposez un énoncé différent pour le problème d'assignations de places dans un mariage, avec des contraintes plus souples, tel que le problème puisse tout le temps avoir une solution.

Plutôt que d'imposer des contraintes sur les placements des personnes sur les tables, on exprime des préférences sur les placements de personnes sur les tables. Pour ce faire, on exprime les contraintes comme avant; c-à-d., une contrainte est une paire de personnes ne devant pas s'asseoir sur la même table ou à une table adjacente; ou au contraire devant s'asseoir sur la même table où à une table adjacente. En plus on donne une fonction de coût, prenant une contrainte comme argument et retournant un nombre réel exprimant le coût lié à la violation de la contrainte; ainsi la fonction exprime le degré de désirabilité de la contrainte. Dorénavant on est sûr d'avoir toujours une solution. Par contre, une solution retournée pourrait violer certaine contraintes.

d. (1 point) Expliquez comment vous formaliseriez le nouveau problème, tel qu'énoncé dans votre réponse au point c, comme un problème de satisfaction de contraintes.

Même formalisation qu'avant, sauf qu'on ajoute en plus la fonction de coût qui étant donnée une contrainte retourne un nombre réel exprimant le coût lié à la violation de la contrainte.

e. (1 point) Proposez une technique (algorithme) pour résoudre le nouveau problème tel que formulé dans votre réponse précédente.

Utiliser *A**.

- *Un nœud est une assignation (partielle) comme dans Backtracking-Search*

  Comme dans Bakktracking-Search, dans un nœud, on assigne une variable à la fois.

- *Étant donné une assignation (partielle), définissons le coût de l'assignation comme étant la somme des coûts des contraintes violées par l'assignation.*
- *Le coût d'une transition entre deux assignations successives est la différence des coûts entre ces transitions.*
- *La fonction heuristique donne l'estimé du coût entre le nœud courant (assignation partielle) et une assignation complète optimale :*

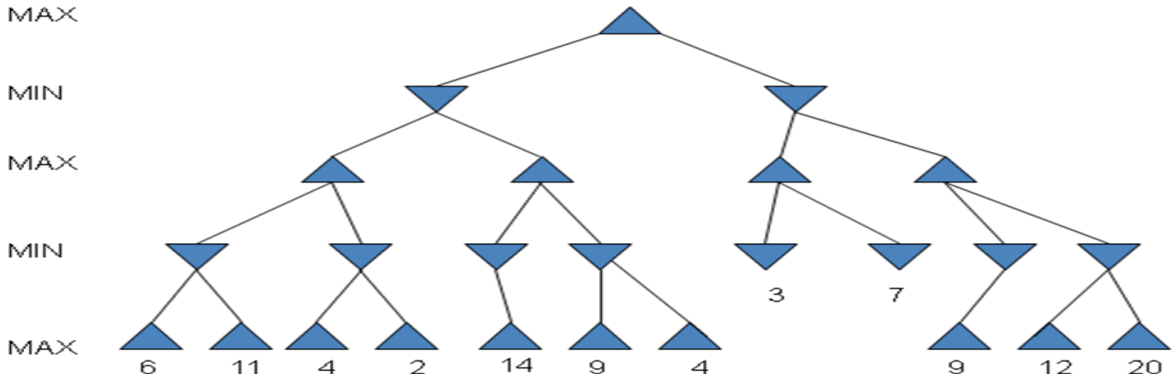
  On pourrait la calculer en estimant le coût de la violation des contraintes par les variables restantes

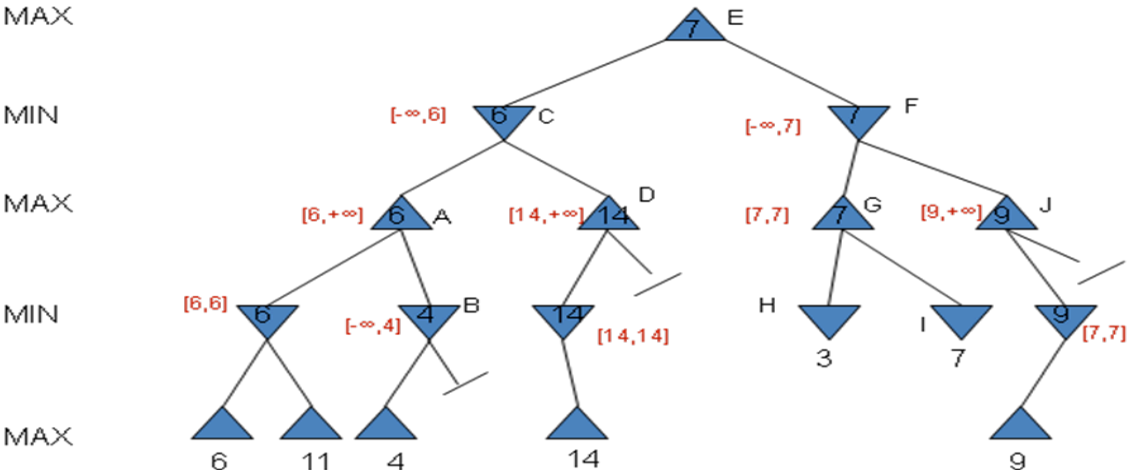
  (c-à-d., les variables non encore présentes dans l'assignation partielle courante).

Il peut y avoir de meilleures fonctions h, mais c'est un véritable défi de les définir.

Question 2 (5 points) – *Alpha-Beta Pruning*

(5 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. Dessinez la partie de l'espace d'états qui serait explorée par l'algorithme *alpha-beta pruning*, en supposant qu'il explore l'espace d'états de la gauche vers la droite. Dessinez seulement les états explorés et les transitions correspondantes. Indiquez, à côté de chaque état exploré, la valeur correspondante à la terminaison de l'algorithme.





Étiquettes : La valeur (v) du nœud, ainsi que la paire [α, β] à chaque nœud, à la terminaison de l'algorithme.

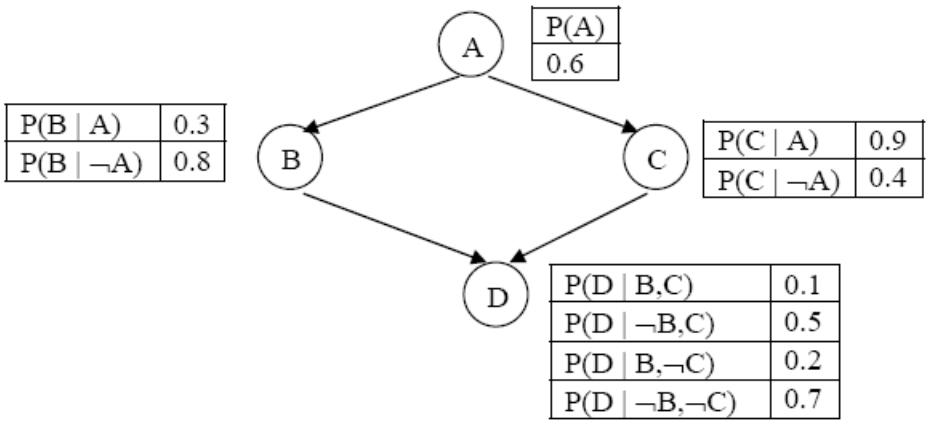
Rappel *:*

- *α : meilleure (plus grande) valeur de MAX jusqu'ici; elle ne décroît jamais; MAX ne considère jamais les nœuds MIN successeurs (en bas de lui) ayant des valeurs plus petites que α.*
- *β : meilleure (plus petite) valeur de MIN jusqu'ici; elle ne croît jamais; MIN ne considère jamais les nœuds MAX successeurs (en bas de lui) ayant des valeurs plus grandes que β.*

A a α = 6 (A ne sera jamais plus petit que 6) *Ainsi, B est α -coupé puisque 4 < α = 6* *C a β = 6 (C ne sera jamais plus grand que 6)* *Ainsi, D est β-coupé puisque 14 > β = 6*	*(La première fois qu'on atteint G, en descendant, on a [α, β] = [-∞,+∞]. En remontant de H, [α, β] = [3,+∞]. En remontant de I, on a [α, β] = [7,+∞]).* *F a β = 7 (F ne sera jamais plus grand que 7)* *Ainsi, J est β-coupé puisque 9 > β = 7* *A la fin, E = 7.*

Question 3 (5 points) – Réseaux bayésiens

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



a. (1 point) Combien de valeurs indépendantes faudrait-il pour stocker la distribution jointe des probabilités?

$2^4 - 1 = 15$.

b. (1 point) Pour ce réseau, est-ce vrai ou faux que P(A,C,D)=P(A)P(C)P(D)? Justifiez votre réponse.

Faux.

Les trois variables ne sont pas indépendantes.
En fait, $P(A,C,D) = \sum_B P(A,B,C,D) = \sum_B P(D|B,C)P(B|A)P(C|A)P(A)$
(Dans la somme on considère B=true (B) et B=False (¬ B))

c. (1 point) Pour ce réseau, est-ce vrai ou faux que P(D|C)=P(D|A,C)? Justifiez votre réponse.

Faux.

A n'est pas indépendant de D sachant C. Ça le serait si on connaissait B et C.

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

Vrai.

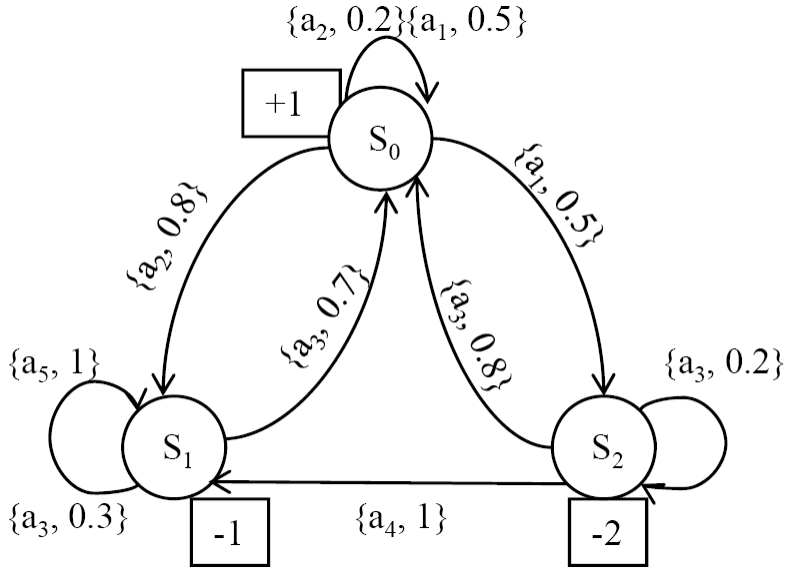
P(D|B,C,A)=P(D|B,C) parce que B et C constituent la couverture de Markov de D.
(ou : parce qu'un nœud est indépendant de ses ancêtres étant donné ses parents).

e. (1 point) Calculez P(¬ C | A, B).

$P(\neg C \mid A, B) = P(\neg C \mid A) = 1 - P(C|A) = 1 - 0.9 = 0.1$

Question 4 (5 points) Planification par les 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 (exprimant une fonction d'utilité).



a. (2 points) En utilisant une approche par processus de décision markoviens, indiquez comment calculer la valeur du plan $\{S_0 \rightarrow a_1, \quad S_1 \rightarrow a_3, \quad S_2 \rightarrow a_4\}$ dans l'état S_2 (en supposant que l'exécution du plan commence dans cet état). Soyez le plus précis possible dans l'espace alloué. Utilisez un facteur d'atténuation (*discount factor*) de 0.2.

Notons v_i la valeur du plan dans l'état s_i et ri la récompense reçue dans l'état s_i.

Par définition, $v_i = r_i + 0.2 \sum_j pr(s_i,a_i,s_j)\, v_j$

Ce qui donne le système d'équations :

$V_0 = 1 + 0.2\,(0.5\, V_0 + 0.5\, V_2)$

$V_1 = -1 + 0.2\,(0.7\, V_0 + 0.3\, V_1)$

$V_2 = -2 + 0.2\,(1\, V_1)$

On obtient la valeur du plan dans l'état S_2 en résolvant ce système d'équations.

b. (1 point) Donnez la valeur du plan au point (a) dans l'état S_2, selon votre formulation.

En résolvant ce système d'équations on a :

$V_0 = 0.868121442$

$V_1 = -0.934535104$

$V_2 = -2.186907021$

Donc la valeur du plan dans l'état S_2 est : -2.186907021

c. (1 point) Dans le tableau suivant, indiquez les valeurs des états à la fin de chacune des trois premières itérations de l'algorithme *value-itération*, en supposant qu'on utilise un facteur d'atténuation (*discount factor*) de 0.2 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 Itération 1	**Valeurs Itération 2**	**Valeurs Itération 3**
S_0	1	0.900	0.898
S_1	-1	-0.920	-0.929
S_2	-2	-1.920	-1.933

d. (1 point) Donnez le plan d'actions (*policy*) résultant de l'itération 3.

Plan :

État	*Action*
S_0	a_1
S_1	a_3
S_2	a_3