Examen Session 1 – Janvier 2020

Durée : 2 heures
Aucun document autorisé

Veuillez rendre sur deux copies séparées les parties CSP (exercices 1 et 4) et règles (exercices 2 et 3)

Exercice 1 : Résolution de CSP 7 pts

Soit le réseau de contraintes (X, D, C) où :

X = {v, w, x, y, z}
D(v) = D(x) = {a, b, c}
D(w) = D(z) = {1, 2, 3, 4}
D(y) = {1, 2, 3, 4, 5}
C = {c_1, c_2, c_3, c_4, c_5, c_6} avec :

c_1

v	w
a	2
a	3
b	1
b	2
b	4

c_2

v	x
a	a
b	a
b	c
c	a

c_3

w	x
1	a
3	a
1	c
2	b

c_4

x	z
a	2
a	3
a	4
b	1

c_5

w	y	z
1	1	1
1	1	2
1	2	2
1	4	1
2	4	1
3	1	1
3	2	3
3	3	2
3	5	3
4	1	1
4	2	1
4	3	2

c_6

v	y
a	1
a	3
a	4
b	1
b	2
b	3
b	5
c	1
c	3

1) Représentez ce réseau par un graphe (biparti d'incidence)

2) Représentez l'exécution de l'algorithme de Backtrack pour la recherche d'**une solution** en utilisant l'ordre alpha-numérique comme ordre d'affectation des variables et valeurs. On attend que vous dessiniez l'arbre de recherche dont les nœuds représentent une assignation ; la racine de l'arborescence représentant l'assignation vide. Chaque niveau de l'arbre est dédié à une variable. Pour faciliter la représentation **on se limitera à indiquer sur chaque nœud la valeur assignée** à la variable correspondant au niveau courant (l'assignation courante étant donc « lue » en remontant le chemin du nœud courant à la racine). Lorsqu'une assignation viole une contrainte, on indique par un **x** que le nœud n'est pas prolongé en **précisant l'une des contraintes violées**.

3) Donnez la solution calculée.

4) Représentez l'exécution de l'algorithme de Forward Checking à la recherche **d'une solution** en utilisant l'heuristique *dom+deg+alpha* (plus petit domaine courant, puis plus petit degré dans le graphe de contraintes, puis ordre alpha-numérique si égalité) pour l'ordre d'application des variables et l'ordre alphanumérique pour les valeurs. On indiquera à chaque nœud de l'arbre la **variable assignée, la valeur choisie et les modifications apportées par propagation aux domaines** des autres variables.

5) Calculez la fermeture arc-consistante de ce réseau.

Exercice 2 : Systèmes à base de règles en logique des propositions — 5 pts

Soit la base de connaissances K = (BF,BR) avec BF = {D,E} et BR contenant les règles suivantes :

R1 : $B \wedge C \wedge F \rightarrow A$
R2 : $C \wedge D \rightarrow B$
R3 : $E \rightarrow C$
R4 : $A \rightarrow F$
R5 : $G \rightarrow A$
R6 : $B \wedge E \rightarrow A$
R7 : $H \rightarrow G$

1) Chaînage avant : donnez BF*, la base de faits saturée par les règles.
2) Chaînage arrière : on cherche à prouver A. Dessinez l'arbre de recherche correspondant à la remontée du graphe ET-OU, en supposant que l'algorithme considère les **règles par numéro croissant** et les **symboles par ordre d'apparition dans l'hypothèse de la règle considérée**. Vous indiquerez sur chaque feuille traitée : *échec*, *déjà échec,* (*appartient à) BF, déjà prouvé, boucle*. Le but A est-il finalement prouvé ?
3) Le littéral $\neg$ G est-il conséquence de BF $\cup$ BR, au sens de la *conséquence logique classique* (c'est-à-dire a-t-on BF, BR $\models \neg$ G) ?
4) Si on considère la négation du monde clos (notée *not* G), à quelle condition considère-t-on que *not* G « découle » de K ? Qu'en est-il ici ?

Exercice 3 : Systèmes à base de règles en logique du premier ordre — 4 pts

Soit la requête conjonctive **Q** = { p(x,y), p(y,z), q(x,u) } où x, y, z et u sont des variables (Q demande donc toutes les valeurs de x, y, z, et u tel que p(x,y) et p(y,z) et q(x,u)).

1) Soit **BF** = { p(a,b), p(b,c), q(a,c), p(d,a) } où a,b,c et d sont des constantes. Quelles sont les réponses à **Q** sur **BF** ?
2) On ajoute **BR** = { **R** : $p(x,y) \wedge p(y,z) \rightarrow q(x,z)$ }. Donnez la *base de faits saturée* par les règles (on ne vous demande pas le détail du calcul)
3) Quelles sont les réponses à Q sur la base de connaissances **(BF, {R})** ?
4) Vous avez programmé un moteur de chaînage avant pour des règles propositionnelles (positives et conjonctives, comme dans l'exercice précédent). On voudrait l'utiliser pour calculer la saturation d'une base de connaissances en logique du premier ordre (où, comme ci-dessus les règles sont positives et conjonctives). Une façon de faire consiste à procéder en 2 temps. On transforme d'abord les règles du premier ordre en règles propositionnelles en remplaçant brutalement chaque variable d'une règle de toutes les façons possibles par une constante apparaissant dans la base de connaissances. On obtient ainsi des règles propositionnelles : l'atome p(a,b) par exemple, où a et b sont des constantes, est vu comme un seul symbole propositionnel, qui apparaît ou non dans la base de faits. On peut alors utiliser un moteur de chaînage avant pour des règles propositionnelles sur la base de connaissance obtenue.
   a. Considérant l'ensemble de constantes {a,b,c,d} provenant de **BF**, combien de règles propositionnelles obtiendrait-on à partir de la règle **R** ? Expliquez.
   b. Donnez une borne supérieure du nombre de règles propositionnelles obtenu à partir d'une base de connaissances du premier ordre.
   c. Pensez-vous que l'on puisse trouver un algorithme polynomial de saturation d'une base en logique du premier ordre ? Expliquez.

Exercice 4 : Modélisation en CSP 4 pts

SAT-Your-Day est un jeu de puzzle, basé sur des réflexions logiques disponible sur votre magasin d'applications préféré (cf. SAT-Your-Day sur Play Store ou App Store). Le jeu consiste, étant donné des caisses remplies de fruits colorés, à supprimer des fruits de ces caisses de telle manière que :

1. Aucune caisse ne se retrouve vide.
2. Il reste exactement un type de fruit par couleur (par exemple, sur la partie ci-après, pour la couleur orange, que les pommes ou que les prunes)

Notez qu'il est possible d'avoir des caisses avec plusieurs fruits dedans. Plus le nombre de boîtes, de couleurs et de fruits par couleur augmente, plus le jeu est compliqué.

Voici une partie de ce jeu, dans laquelle on trouve 7 caisses remplies de 16 fruits de 4 couleurs (**O**range, **R**ouge, **V**ert, v**I**olet) ayant 6 types différents (**C**itron, fram**B**oise, **F**raise, pastè**Q**ue, po**M**me, pr**U**ne). Tous les types de fruits ne sont pas disponibles dans chaque couleur !

Modélisez la résolution de ce jeu comme un problème de satisfaction de contraintes, c'est-à-dire proposez un réseau de contraintes (X,D,C) tel qu'une solution à ce réseau soit une solution à cette partie de jeu. Il n'est pas demandé de résoudre cette partie mais uniquement de fournir le réseau de contraintes.