Université des Sciences et de la Technologie Houari Boumediène
Faculté d'Electronique et d'Informatique
Département d'Informatique
LMD Master 2ème Année ''Systèmes Informatiques Intelligents'' 2011/2012
Module ''Programmation par Contraintes''
Date : 02/01/2012

Corrigé de l'interrogation
<Corrigé préparé par le responsable du module, Mr ISLI>

Exercice 1 (5 points=<1,5 ; 1,5 ; 2>) :
On considère le CSP binaire discret P=(X,D,C) suivant :

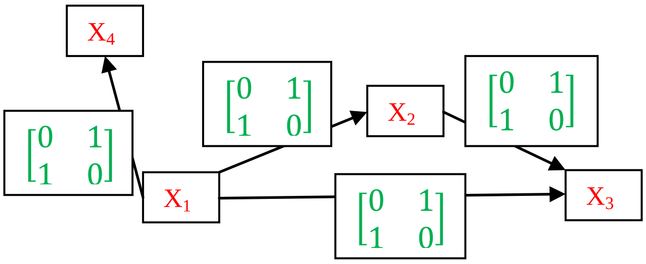
- $X = \{X_1, X_2, X_3, X_4\}$
- $D(X_1)=D(X_2)=D(X_3)= D(X_4)=\{7,9\}$
- $C = \{c_1, c_2, c_3, c_4\}$ avec
    - $c_1 : X_1 \neq X_2$
    - $c_2 : X_1 \neq X_3$
    - $c_3 : X_2 \neq X_3$
    - $c_4 : X_1 \neq X_4$

1) Donnez une représentation matricielle de P.
2) Montrez, en utilisant les définitions vues en cours, sans l'utilisation d'un quelconque algorithme de consistance d'arc, que le CSP P est consistant d'arc.
3) Le CSP est-il consistant ? s'il ne l'est pas, l'algorithme de consistance de chemin PC2 en détecte-il l'inconsistance ?

Solution :

1) Une représentation graphique de P (qui n'est pas unique) est le graphe orienté étiqueté $G_P=(V,E,l)$, dont les sommets sont les variables de P, et les arcs les paires (X_i,X_j) telles que i<j et il existe une contrainte de P entre X_i et X_j :

- V=X
- $E=\{(X_1,X_2),(X_1,X_3),(X_1,X_4),(X_2,X_3)\}$
- L'étiquette d'un arc (X_i,X_j), $l(X_i,X_j)$, est l'intersection des matrices $M_k(X_i,X_j)$ représentant les relations $R_k(X_i,X_j)$ associées aux contraintes c_k portant sur X_i et X_j
    - $l(X_1,X_2)=M_1(X_1,X_2)=\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$
    - $l(X_1,X_3)=M_2(X_1,X_3)=\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$
    - $l(X_2,X_3)=M_3(X_2,X_3)=\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$
    - $l(X_1,X_4)=M_4(X_1,X_4)=\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$



Une représentation matricielle de P est la matrice 4x4 M_P de matrices booléennes 2x2 :

a. Il n'y a pas de contraintes unaires : $M_P[i,i]=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$ (la matrice identité 2x2)

b. Pour tous i et j vérifiant i<j et (X_i,X_j) arc de G_P : $M_p[i,j]= I(Xi,Xj)$ et $M_P[j,i]=(M_P[i,j])^t$ (la trasnposée de $M_P[i,j]$)

c. Pour tous i et j vérifiant i<j et (X_i,X_j) n'est pas arc de G_P : $M_P[i,j]=M_P[j,i]=\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}$ (la relation universelle 2x2)

$$M_P=\begin{bmatrix} \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} & \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} & \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} & \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \\ \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} & \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} & \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} & \begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} \\ \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} & \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} & \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} & \begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} \\ \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} & \begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} & \begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix} & \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \end{bmatrix}$$

2) Le domaine de chacune des variables est {7,9} et les lignes de chacune des matrices booléennes 2x2 de M_P ont chacune au moins un 1. Donc la définition même de la consistance d'arc d'un CSP est vérifiée : pour toute paire (X_i,X_j) de variables, pour toute valeur v_i du domaine de X_i, il existe une valeur v_j du domaine de X_j, telle que l'instanciation $(X_i=v_i,X_j=v_j)$ satisfait toutes les contraintes portant sur X_i et X_j.

3) Le CSP P, d'après la réponse à la question précédente, est consistant d'arc. Le CSP P, cependant, est trivialement inconsistant (l'inconsistance vient du triplet (X_1,X_2,X_3) : on ne peut pas colorier les nœuds d'un triangle avec seulement deux couleurs sans que le coloriage associe la même couleur à deux nœuds adjacents). Si on devait appliquer un algorithme de consistance d'arc à P, il ne détecterait pas l'inconsistance : il laisserait le CSP inchangé. L'algorithme de consistance de chemin PC2 appliqué à P initialiserait la file Q aux paires (Xi,Xj) vérifiant i≤j et il existe une contrainte sur les variables Xi et Xj : Q={(X1,X2),(X1,X3),(X1,X4),(X2,X3)}. Ici, il faut aller droit au but : on veut montrer que PC2 détecte l'inconsistance de P, et l'inconsistance vient du triplet (X1,X2,X3) : aucune solution du sous-CSP sur les variables X1 et X2 ne peut être étendue à la variable X3 : il faut donc faire prendre à PC2 la paire (X1,X2) de la file : Q devient {(X1,X3),(X1,X4),(X2,X3)}. Pour chacune des variables X_k vérifiant $\neg(X_1=X_2=X_k)$ (donc

toutes les quatre variables), PC2 met à jour éventuellement les étiquettes sur les arcs voisins (X_1,X_k) et (X_k,X_2). Pour k=3, l'opération de consistance de chemin $M_P[1,3]=M_P[1,3]\cap M_P[1,2]°M_P[2,2]°M_P[2,3]$ détecte la matrice zéro qui signifie inconsistance :

$M_P[1,3]=M_P[1,3]\cap M_P[1,2]°M_P[2,2]°M_P[2,3]$

$= M_P[1,3]\cap M_P[1,2]°M_P[2,3]$ car $M_P[2,2]$ matrice identité (neutre pour la composition)

$$= M_P[1,3]\cap \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} ° \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$= M_P[1,3]\cap \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cap \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Exercice 2 (6 points=<1 ; 3 ; 2>) :

Soit P un CSP binaire discret donné par sa représentation matricielle M_P. Vous supposerez que P a n variables $X_1,\ldots,X_n$ et que toutes les variables ont le même domaine $dom=D(X_1)=\ldots=D(X_n)$, de taille m. On notera par $M_P[i,j]$ l'élément (i,j) de la matrice M_P.

1) Une instanciation de P est un élément de quel ensemble ? Que représente ledit ensemble pour le CSP P ?
2) Donnez une fonction booléenne **sol** testant si une instanciation est solution de P.
3) Donnez une fonction booléenne **GET** testant si P admet une solution.

Solution :

1) Une instanciation de P est un élément du produit cartésien $D(X1)x…xD(Xn)=(dom)^n$. L'ensemble $(dom)^n$ est l'espace de recherche de P.
2) On suppose le domaine commun donné sous forme d'un tableau $dom=[a_1,\ldots,a_m]$. Une instanciation (e1,…,en) sera représentée par le tableau I=[e1,…,en]

```
Booléen sol(I,M){
    pour i=1 à n faire
            j=1
            tant que I[i]≠dom[j] faire j++ fait
            indice[i]=j
    fait    //cette première partie de la fonction met les indices des éléments
            //de l'instanciation (e1,…,en) dans un tableau indice
    pour i=1 à n faire
            pour j=1 à n faire
                    si M[i,j][indice[i],indice[j]]=0 alors retourner FAUX finsi
            fait
    fait
    retourner VRAI
    }
```

3) booléen GET(I,p,M){

si p=n (toutes les variables de X sont instanciées) alors

si sol(I,M) alors retourner VRAI sinon retourner FAUX finsi

sinon

T=I //équivalent à la boucle : pour j=1 à p faire T[j]=I[j] fait

i=p+1 //on passe à la prochaine variable non encore instanciée

pour j=1 à m faire //j parcourt le domaine commun des variables

T[i]=dom[j] //on instancie Xi avec le jème élément du domaine

si GET(T,i,M) alors retourner VRAI finsi

fait

retourner FAUX

finsi

}

Appel de la fonction GET : GET(INST,0,M), avec INST tableau de taille n utilisé ici pour représenter une instanciation, et M représentation matricielle d'un CSP P dont on veut savoir s'il est consistent.

Exercice 3 (4 points=<2 ; 2>) :

1. On considère un TCSP P dont l'ensemble des contraintes inclut $\{c_1 : (X_2\text{-}X_1)\in]-\infty,5]\cup[30,80], c_2 : (X_3\text{-}X_2)\in[3,10]\}$. Quelle est la contrainte sur les variables extrêmes X_1 et X_3 inférée par composition de c_1 et c_2 ? Expliquez.

4) On considère un CSP P de directions cardinales, dont l'ensemble des contraintes inclut $\{c_1 : \{NW,SE\}(X_1,X_2), c_2 : \{S\}(X_2,X_3)\}$. Quelle est la contrainte sur les variables extrêmes X_1 et X_3 inférée par composition de c_1 et c_2 ? Expliquez.

Solution :

1) La contrainte c1 signifie

$-\infty < x2\text{-}X1 \leq 5$ (c1-1)

ou

$30 \leq X2\text{-}X1 \leq 80$ (c1-2)

La contrainte c2 à son tour signifie

$3 \leq X3\text{-}X2 \leq 10$ (c2)

L'addition membre à membre de c1-1 et c2 puis de c1-2 et c2 donne :

$-\infty < x3\text{-}X1 \leq 15$ (c3-1)

ou

$33 \leq X3\text{-}X1 \leq 90$ (c3-2)

La contrainte sur les variables extrêmes X1 et X3 inférée par composition de c1 et c2 est donc c3 : $(X3\text{-}X1)\in]-\infty,15]\cup[33,90]$

2) La contrainte c1 signifie

$\{NW\}(X_1,X_2)$ (c1-1)

ou

$\{SE\}(X_1,X_2)$ (c1-2)

La composition de c1-1 et c2 puis de c1-2 et c2 donne :

{SW,W,NW}(X1,X3) (c3-1)

ou

{SE}(X1,X3) (c3-2)

La contrainte sur les variables extrêmes X1 et X3 inférée par composition de c1 et c2 est donc c3 : {SW,W,NW,SE}(X1,X3)

Exercice 4 (5 points=<2 ; 3>) :

1) En quoi consiste un programme Prolog ?
2) Ecrire un programme Prolog calculant la fermeture réflexive-transitive d'un graphe orienté : la fermeture réflexive-transitive d'un graphe orienté G=(X,R), avec X l'ensemble des sommets et R l'ensemble des arcs, est le graphe orienté G*=(X,R*) tel que R* est la fermeture réflexive-transitive de R (la plus petite relation réflexive et transitive contenant R).

Solution :

1) Un programme Prolog consiste en un ensemble de paquets, un paquet étant un ensemble de clauses de Horn avec le même atome de tête. Un programme Prolog est interprété conjonctivement (conjonction des paquets le composant). Un paquet, lui, est interprété disjonctivement (disjonction des clauses le composant).
2) Le prédicat reflexive(L1,L2), où L1 est une liste des éléments d'un ensemble S, calcule une liste L2 des paires d'éléments de S (la liste L2 est utilisée ici pour représenter la plus petite relation réflexive sur S). Le prédicat graphe(S,R) teste que les arcs représentés par la liste R sous forme de paires ont bien leurs sommets dans S. Le prédicat ferm_refl_trans(S,R,Rstar) calcule la fermeture réflexive-transitive (S,Rstar) du graphe (S,R). Le prédicat composition(L1,L2,L3) calcule dans L3 la composition de L1 et L2 : L3 est la liste des éléments [X,Z] pour les quels il existe Y tel que [X,Y] et [Y,Z] appartiennent, respectivement, à la liste L1 et à la liste L2. Le programme procède comme suit : la fermeture réflexive-transitive d'un graphe est la fermeture transitive de la fermeture réflexive du graphe. Il est vrai que, et c'est plus économique, qu'on peut aussi procéder comme ceci : la fermeture réflexive-transitive d'un graphe est la fermeture réflexive de la fermeture transitive !

```
element(X,[X|R]).
element(X,[Y|R]) :- element(X,R).
graphe(S,[]).
graphe(S,[[X,Y]]):-element(X,S),element(Y,S).
graphe(S,[A|L]):-graphe(S,[A]),graphe(S,L).
ferm_refl_trans(S,R,Rstar):-graphe(S,R),ferm_refl(S,R,R1),ferm_trans(S,R1,Rstar).
ferm_refl(S,R1,R2):-reflexive(S,R3),append(R3,R1,R2).
```

```
ferm_trans(S,R,R):-composition(R,R,R).
ferm_trans(S,R1,R2):-composition(R1,R1,R3),ferm_trans(S,R3,R2).
composition([],_,[]).
composition(_,[],[]).
composition([[X,Y]],[[Y,Z]],[[X,Z]]).
composition([[X,Y]],[[T,Z]],[]).
composition([A],[B|L1],L2) :-
    composition([A],[B],L3),composition([A],L1,L4),append(L3,L4,L2).
composition([A|L1],L2,L3) :-
    composition([A],L2,L4),composition(L1,L2,L5),append(L4,L5,L3).
reflexive([],[]).
reflexive([X],[[X,X]]).
reflexive([X|L],R):-reflexive([X],R1),reflexive(L,R2),append(R1,R2,R).
```