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'' 2010/2011
Module ''Programmation par Contraintes''
Date : 13/01/2011

Corrigé de l'interrogation

Exercice 1 :
On considère le CSP discret P=(X,D,C) suivant :

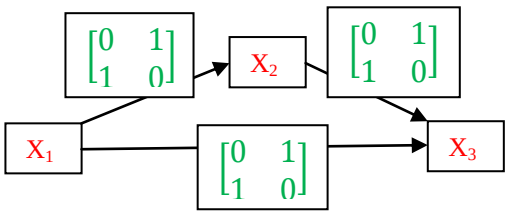
- $X = \{X_1, X_2, X_3\}$
- $D(X_1)=D(X_2)=D(X_3)=\{100,200\}$
- $C = \{c_1, c_2, c_3\}$ avec
  - $c_1 : X_1 \neq X_2$
  - $c_2 : X_1 \neq X_3$
  - $c_3 : X_2 \neq X_3$

1) Donnez une représentation graphique de P.
2) Donnez une représentation matricielle de P.
3) 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.
4) Que pouvez-vous déduire de la réponse à la question précédente ?

Solution :

1) Une représentation graphique de P (qui n'est pas unique) est le graphe orienté étiqueté $G_P=(V,E,e)$, 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
- $E=\{(X_1,X_2),(X_1,X_3),(X_2,X_3)\}$
- L'étiquette d'un arc (X_i,X_j), e_{ij}, 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
  - $e_{12}=M_1(X_1,X_2)=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$
  - $e_{13}=M_2(X_1,X_3)=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$
  - $e_{23}=M_3(X_1,X_3)=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$



2) Une représentation matricielle de P est la matrice 3x3 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 :$M_p[i,j]=e_{ij}$ et $M_P[j,i]=(M_P[i,j])^{-1}$ (l'inverse 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)

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

3) Le domaine de chacune des variables est {100,200} 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.
4) Le CSP P, d'après la réponse à la question précédente, est consistant d'arc. Le CSP P, cependant, est trivialement inconsistant (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. La consistance d'arc est donc incomplète même pour les CSP dont les domaines des variables ont une taille (nombre d'éléments) ne dépassant pas deux. De plus, un algorithme de résolution comme le Look_Ahead, qui, au niveau de chaque nœud de l'arbre de recherche, instancie une variable puis applique un filtrage par consistance d'arc au CSP résultant, ne peut pas faire en sorte que l'instanciation soit par groupe de deux valeurs du domaine de la variable à instancier : une telle politique d'instanciation pourrait générer, après avoir instancié toutes les variables, un CSP raffinement du CSP de départ qui est consistant d'arc mais inconsistant ; et un tel algorithme ne serait donc pas complet, contrairement à un algorithme de résolution basé sur une instanciation des variables avec une valeur du domaine, dont la complétude est justifiée par la complétude de la consistance d'arc pour les CSP dont les domaines des variables sont tous de taille 1 (singletons) !

Exercice 2 :

On considère le CSP qualitatif de directions cardinales P donné par les contraintes c_1, c_2, c_3 et c_4 suivantes :

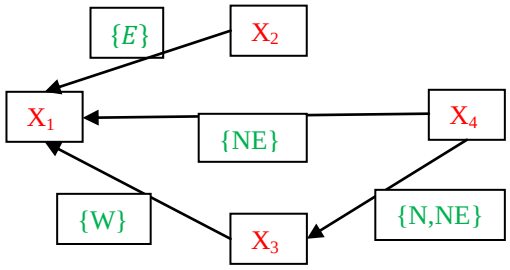
- c_1: Béjaia est à l'est d'Alger
- c_2 : Oran est à l'ouest d'Alger
- c_3 : Le bâteau est au nord-est d'Alger
  - satellite 1 à l'instant t
- c_4 : Le bâteau est au nord ou au nord-est d'Oran
  - satellite 2 au même instant t

On associera aux villes Alger, Béjaia et Oran les variables X_1, X_2 et X_3, respectivement ; et au bâteau la variable X_4.

1) Donnez une représentation graphique de P.
2) Donnez une représentation matricielle de P.
3) Appliquez l'algorithme de consistance de chemin PC3 au CSP P.
4) Expliquez le principe de fonctionnement de l'algorithme de recherche Look_Ahead sur un CSP qualitatif de directions cardinales.
5) Utilisez l'algorithme Look_Ahead pour montrer si le CSP ci-dessus est consistant.

Solution :

1)



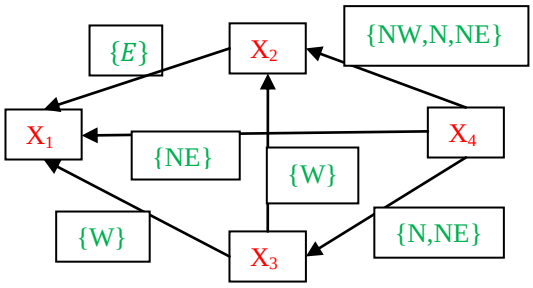
2) M_P=

	X1	X2	X3	X4
X1	{EQ}	{W}	{E}	{SW}
X2	{E}	{EQ}	?	?
X3	{W}	?	{EQ}	{S,SW}
X4	{NE}	?	{N,NE}	{EQ}

3) **Application de l'algorithme de consistance de chemin PC3 au CSP P :**
   a. Initialisation de la file F : F={(X_2,X_1),(X_3,X_1),(X_4,X_1),(X_4,X_3)}
   b. On prend l'arc (X_2,X_1) de la file pour propagation : la file devient F={(X_3,X_1),(X_4,X_1),(X_4,X_3)}
   c. Pour tous les k dans {3,4} : $e_{k1}=e_{k1}\cap(e_{k2}\circ e_{21})$ et $e_{k2}=e_{k2}\cap(e_{k1}\circ e_{12})$
      i. $e_{31}=e_{31}\cap(e_{32}\circ e_{21})$= {W}∩(?°{E})={W}
      ii. $e_{32}=e_{32}\cap(e_{31}\circ e_{12})$= ?∩({W}°{W})={W}
      iii. (X_3,X_2) modifié ; la file devient F={(X_3,X_1),(X_4,X_1),(X_4,X_3),(X_3,X_2)}
      iv. $e_{41}=e_{41}\cap(e_{42}\circ e_{21})$= {NE}∩(?°{E})={NE}

v. $e_{42}=e_{42}\cap(e_{41}\circ e_{12})$ = ?$\cap$({NE}°{W})={NW,N,NE}

vi. (X_4,X_2) modifié ; la file devient F={(X_3,X_1),(X_4,X_1),(X_4,X_3),(X_3,X_2),(X_4,X_2)}



d. On prend l'arc (X_3,X_1) de la file pour propagation : la file devient F={(X_4,X_1),(X_4,X_3),(X_3,X_2),(X_4,X_2)}

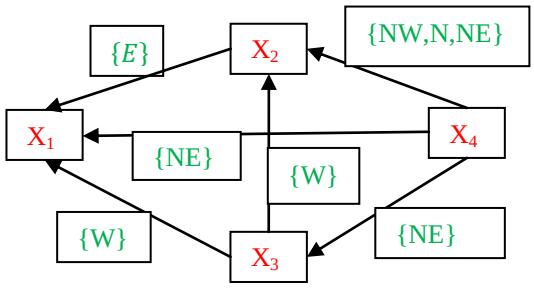
e. Pour tous les k dans {2,4} : $e_{k3}=e_{k3}\cap(e_{k1}\circ e_{13})$ et $e_{k1}=e_{k1}\cap(e_{k3}\circ e_{31})$

i. $e_{23}=e_{23}\cap(e_{21}\circ e_{13})$= {E}$\cap$({E}°{E})={E}

ii. $e_{21}=e_{21}\cap(e_{23}\circ e_{31})$= {E}$\cap$({E}°{W})={E}

iii. $e_{43}=e_{43}\cap(e_{41}\circ e_{13})$={N,NE}$\cap$({NE}°{E})={NE}

iv. (X_4,X_3) modifié mais il est déjà dans la file



v. $e_{41}=e_{41}\cap(e_{43}\circ e_{31})$= {NE}$\cap$({NE}°{W})={NE}

f. La propagation des quatre autres arcs de la file ne modifie aucunement le CSP (travail laissé à l'étudiant) : le CSP a donc été rendu consistant de chemin

4) **Principe de fonctionnement de l'algorithme Look_Ahead :**

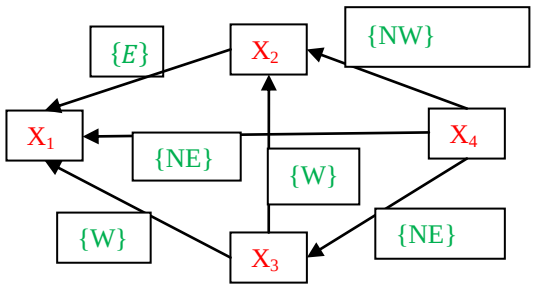
L'algorithme Look_Ahead appliqué à un CSP qualitatif P de directions cardinales est un algorithme de recherche récursive d'un raffinement atomique (donc convexe) chemin-consistant de P, si P admet un tel raffinement ; si un tel raffinement n'existe pas, l'algorithme retourne que le CSP de départ est inconsistant. Un raffinement atomique est un raffinement dont chaque paire de variables est reliée par une relation atomique. Au niveau de chaque nœud de l'arbre de recherche, l'algorithme teste s'il y a encore des arcs disjonctifs au niveau du CSP étiquetant le nœud. Si le test est négatif, tous les arcs sont atomiques et le raffinement étiquetant le nœud, étant atomique et consistant de chemin, il est consistant. Si le test est positif, un arc disjonctif est choisi pour instanciation et la consistance de chemin est appliquée au CSP résultant. Si la

relation vide est détectée par le filtrage par consistance de chemin, un retour arrière est fait à l'arc disjonctif le plus récemment instancié et présentant des relations atomiques non encore choisies. S'il n'y a plus d'arcs disjonctifs présentant des choix non encore effectués, l'algorithme termine avec échec (inconsistance). Si la relation vide n'est pas détectée par le filtrage par consistance de chemin, de deux choses l'une :

- Il n'y a plus d'arc disjonctif, auquel cas le raffinement étiquetant le nœud, étant atomique et consistant de chemin, il est consistant
- Il y a encore des arcs disjonctifs : réitérer le processus de choix d'un arc disjonctif à instancier

La complétude de l'algorithme de Look_Ahead appliqué à un CSP qualitatif de directions cardinales est justifiée par un résultat connu dans la littérature disant que si un tel CSP est atomique et consistant de chemin alors il est consistant.

5) La consistance de chemin appliqué au CSP P (question 3) peut être vue comme un pré-traitement de l'algorithme Look_Ahead. Le CSP P, après ce pré-traitement, présente un seul arc disjonctif, l'arc (X_4,X_2) étiqueté par {NW,N,NE}. L'algorithme Look_Ahead choisit donc cet arc pour instanciation : il l'instancie avec une des trois relations atomiques constituant son étiquette puis il applique un filtrage par consistance de chemin. On suppose que l'arc (X_4,X_2) est instancié avec {NW}. La consistance de chemin est appliquée au raffinement résultant, avec une file initialisée à l'arc instancié, (X_4,X_2). La consistance de chemin laisse le raffinement inchangé (je laisse la vérification à l'étudiant): le raffinement était déjà consistant de chemin ; ce raffinement, étant atomique et consistant de chemin, il est consistant



Exercice 3 :

1) En quoi consiste un programme Prolog ?
2) Ecrire un programme Prolog testant si un élément est présent au moins deux fois dans une liste.

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)
```
element_2(X,[X|R]) :-element_1(X,R).
element_2(X,[Y|R]) :- element_2(X,R).
element_1(X,[X|R]).
element_1(X,[Y|R]) :- element_1(X,R).
```