Charrier Florent
Jauffret Julien

26/04/2019

Algorithme bigMAC
Structure de données

Ce document est une présentation des structure de données utilisé pour representer les CSP dans notre projet.

On a donc 4 structures :
-Domain
-Constraint
-Constraint_mat
-CSP

Structure Domain
Cette structure represente l'ensemble des domaines du CSP.
Elle est composée de deux entiers et d'une matrice de booleen : le nombre de variables et la taille max du domaine.
La matrice est rectangulaire et de taille nb_variables X |domaine le plus grand|.
Elle contient des booleen
Ainsi matrice[Xi][Yj] = 1 signifie que le domaine de la variable Xi contient la valeur Yj .
Exemple :

N/D	0	1	2	3	...	m
X0	0	1	1	1		1
X1	0	0	1	0		1
...		1	1	0		1
Xn	0	0	0	1		1

Structure Constraint
Cette structure permet de representer une contrainte.
Elle est composé d'un entier taille max domaine et d'une matrice "relations".
La matrice relations est une matrice rectangulaire de taille |D(Xi)| x |D(Xj)] (carré dans notre cas car on utilise le même domaine partout pour simplifier un peu).
C'est une matrice de booleen qui permet de définir quels sont les couples de valeur autorisé par la contrainte.
Ainsi, relation[i][j] = 1 indique que le couple de valeur (i,j) affecté aux deux variables de cette contrainte, ne viole pas la contrainte.

Structure Constraint_mat
Cette structure permet de representer l'ensemble des contraintes.
Elle est composé d'un entier nb_var et d'une matrice carre nb_varXnb_var de struct Constraint "constraint_matrix".
Ainsi, chaque case matrix[i][j] peut donc avoir deux valeur :
-NULL si il n'y a pas de contrainte entre les variables i et j
-Un struct Constraint

Structure CSP
Il s'agit simplement d'une structure "valise" pour representer le CSP.
On a donc une liste de variables, un struct Constraint_mat pour les contraintes et un struct Domain pour representer le domaine.