Université Paris 7 - Master 1 Informatique - Programmation logique par contraintes

Examen du 6 janvier 2012 - Durée : 2 heures

Informations : Tous les documents sont autorisés. Le barème est donné à titre indicatif et peut être modifié.

Exercice 1 (**4 points**)
Considérez la contrainte $X < Y \ \wedge \ Y < Z \ \wedge \ Z \leq X$ avec domaines $D(X) = D(Y) = D(Z) = [1...1000]$.

- Est-ce que cette contrainte a une solution ?
- Rendez la contrainte borne-consistante en considérant une par une les contraintes simples. Faites uniquement les premiers 6 pas. Combien de pas sont nécessaires avant de s'arrêter ?
- Si on demande à GPROLOG `?- X #< Y, Y #< Z, Z #=< X.` la réponse est donnée après environ 16 secondes ! Pourquoi ?
- Dans YAP on fait d'abord `?- use_module(library(clpr)).` Ensuite on demande `?- {X < Y, Y < Z, Z =< X}.` La réponse est donnée quasiment instantanément. Pourquoi ?

Exercice 2 (**4 points**)
Minimiser Z par rapport à $X \geq 0, Y \geq 0, Z \geq 0$ et

$$\begin{aligned} 2 * X - 2 * Y + 3 * Z &\leq 1 \\ 2 * X + 2 * Y + 3 * Z &= 2 \\ 5 * X - 6 * Y &= 3 \end{aligned}$$

Indication : Si vous pivotez une variable, il est conseillé de commencer avec X.

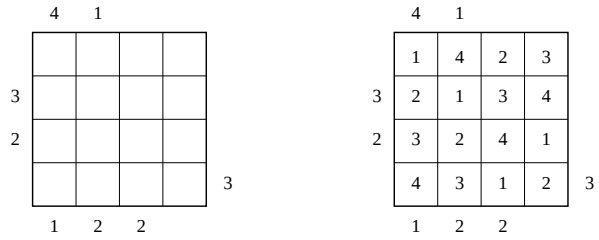
Exercice 3 (**4 points**)
Un nutritionniste veut mettre en place un régime alimentaire à la fois équilibré et le moins cher possible. On suppose que les seuls aliments disponibles sont les suivants :

aliment	taille d'une portion	énergie (kcal)	protéine (gramme)	calcium (mg)	prix (Euros/portions)	limite (portions/jour)
flocons d'avoine	28g	110	4	2	0.3	4
lait	221ml	160	8	285	0.9	8
poulet	100g	205	32	12	2.4	3
oeufs	2	160	13	54	1.3	2
tarte au citron	170g	420	4	22	2	2
boeuf-carotte	260g	260	14	80	1.9	2

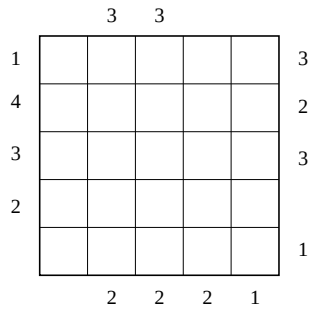
Le régime alimentaire doit contenir au moins 2000 kcal d'énergie, 55g de protéine et 800 mg de calcium. Une certaine variété est imposé (car qui voudra manger 10 portions de boeuf-carotte par jour ?) et pour chaque aliment un nombre limite de portions par jour est donc fixé (le nombre de portions/jour peut être une fraction). Le nutritionniste se pose la question : Quel est le régime alimentaire satisfaisant qui est le **moins** cher ?

- Aidez le nutritionniste en **modélisant** le problème comme un CSP. Il vous est demandé de **modéliser** et non pas de **résoudre** le problème.

Exercice 4 (**8 points**)
Des gratte-ciel sont construits sur une grille NxN. Chaque gratte-ciel a un nombre d'étages correspondant à sa taille (de 1 à N). Dans chaque colonne et chaque ligne chaque taille apparaît exactement une fois. Les nombres sur les bords de la grille indiquent combien de gratte-ciel peuvent être vus en regardant de ce point vers la ligne ou la colonne correspondante. Un gratte-ciel est visible, si **tous** les gratte-ciel devant lui sont plus petits. Voici l'exemple d'une grille et de sa solution :



Par exemple, dans la deuxième ligne du bas en regardant de la gauche on peut voir 2 gratte-ciel : celui d'hauteur 3 et celui d'hauteur 4. Voici un deuxième exemple d'une grille (5x5) :



- Donnez un programme en GPROLOG qui résout le deuxième exemple. Il vous est demandé d'écrire un programme pas de résoudre vous-même le problème.
  Indications : Vous pouvez utiliser le prédicat prédéfini `fd_cardinality(+List, ?Count)` où `List` est une liste de contraintes et `Count` le nombre de contraintes de `List` satisfaites. Dans les contraintes, vous pouvez utiliser la conjonction `#/\`. Par exemple, `fd_cardinality([3 #< 4 #/\ 4 #< 5, 3 #< 2],1)` est vrai.
- Donnez un programme en GPROLOG qui résout un problème de gratte-ciel quelconque. Pour cela, donnez d'abord une description de la représentation du problème (taille de la grille, etc.).
  Indication : Vous pouvez supposer qu'il existe un prédicat `transpose(+L,?T)` qui étant donné une liste de listes `L` représentant une grille renvoie dans `T` une liste de listes avec colonnes et lignes inversés. Par exemple, `transpose([[1,2,3],[4,5,6],[7,8,9]],T)` renvoie `T = [[1,4,7],[2,5,8],[3,6,9]]`. Vous pouvez également utiliser `reverse(+L,?R)` qui renverse une liste `L`. Par exemple, `reverse([1,2,5],R)` donne `R=[5,2,1]`.
  - Commencez par écrire un prédicat qui génère une grille NxN de variables de la forme `[[...],[...],...,[...]]`