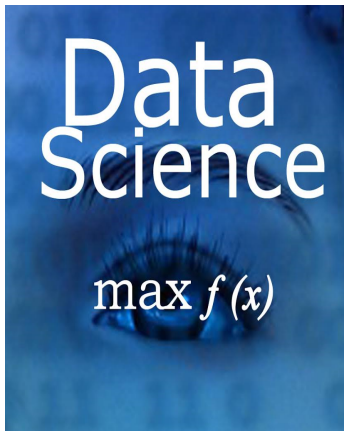
Programmation par Contraintes

GNU Prolog en résumé

Maria Malek

2 janvier 2017



Modélisation de variables

- Domaines finis.
- Représentation par intervalles : *fd_max_integer*/1.
- Représentation par valeurs : *fd_vector_max*/1.
- Suite à une contrainte : passage à la représentation par valeurs.
- exemple :
  - Hypothèse $vector\_max = 127$:
  - $X\# =< 512$, le domaine est 0..512.
  - $X\#\backslash = 10$, le domaine devient 0..9 : 11..127.
  - $X\# =< 100$,le domaine devient 0..9 : 11..100

Les domaines des variables

- Les paramètres :
  - *fd_max_integer*(*N*) est vraie ssi N est la valeur maximale.
  - *fd_vector_max*(*N*).
  - *fd_set_vector_max*(*N*).
- Initialisation des domaines
  - *fd_domain*(*Vars*, *Lower*, *Upper*) : initialise les domaines de Vars.
  - *fd_domain_bool*(*Vars*)
  - *fd_domain*(*Vars*, *Values*).

Test des types & de valeurs

- Les tests de types :
  - *fd_var*(*T*) est vraie ssi T est une FD variable.
  - *non_fd_var*(*T*).
- Informations sur les valeurs.
  - *fd_min*(*X*, *N*) est vraie ssi *N* est la valeur minimale du domaine courant de X.
  - *fd_max*(*X*, *N*), *fd_size*(*X*, *N*)
  - *fd_dom*(*X*, *Values*) est vraie ssi Values correspond aux valeurs du domaine courant de X.
  - *fd_has_vector*(*X*) est vraie ssi X utilise une représentation par valeurs.
  - *fd_use_vector*(*X*)

Les contraintes arithmétiques

- Variables & Expressions dans le domaine *integer*.
- Les opérations :+,-,*,/,**,$min(E_1, E_2)$, $max(E_1, E_2)$,$dist(E_1, E_2)$, $E_1//E_2$,$E_1 \text{rem} E_2$
- $\#=$,$\#\backslash=$,$\#<$,$\#>$,$\#=<$,$\#>=$
- $\#=\#$,$\#\backslash=\#$,$\#<\#$,$\#>\#$,$\#=<\#$,$\#>=\#$
- #pred : on utilise la borne consistance.
- #pred# : on utilise l'arc-consistance.
- exemple :
  - $fd\_domain(X, 1, 8), fd\_domain(Y, 2, 7), X\# = 2 * Y$.
  - $fd\_domain(X, 1, 8), fd\_domain(Y, 2, 7), X\# = \#2 * Y$.

Les contraintes booléennes

- Variables & Expressions dans le domaine *booléen* : {0,1}.
- Les opérations :$\#\backslash E$,
  $E_1\#$ <=> $E_2, E_1\#\backslash$ <=> $E_2, E_1\#\#E_2, E_1\#h$ ==> E_2,
  $E_1\#\backslash$ ==> $E_2, E_1\#/\backslash E_2, E_1\#\backslash/\backslash E_2, E_1\#\backslash/E_2, E_1\#\backslash\backslash/E_2$.
- Exemple :
  $fd\_domain\_bool([X, Y])$, $X\#/\backslash Y$.

Les contraintes symboliques

- List : une liste de valeurs, L : une liste de variables.
- $fd\_all\_different(L)$.
- $fd\_element(I, List, X)$, $fd\_element\_var(I, L, X)$
- $fd\_atmost(N, L, V)$: Au plus N variables de L sont égaux à la valeur V.
- $fd\_atleast(N, L, V)$,$fd\_exactly(N, L, V)$
- $fd\_relation(Relations, Vars)$
- Exemple : $and(X, Y, Z)$:
  $-fd\_relation([[0, 0, 0], [0, 1, 0], [1, 0, 0,], [1, 1, 1]], [X, Y, Z])$

L'affectation des variables

- *fd_labeling*(*Vars*, *Options*) affecte les variables en utilisant l'algorithme backtrack.
- Options est une liste :
  - *variable_method*(*V*) : standard, *first_fail*, *most_constrained*, smallest, largest, *max_regret*, random.
  - recorder(true/false).
  - *value_method*(*V*) : min, max, middle, bounds, random
  - *backtracks*(*B*)
- *fd_labeling*(*Vars*) est équivalent à *fd_labeling*(*Vars*, [])
- *fd_labelingff*(*Vars*) est équivalent à *fd_labeling*(*Vars*, [*variable_method*(*ff*)])

Les contraintes d'optimisation

- $fd\_minimize(But, X)$ Appelle le but But afin de trouver une valeur qui minimize X.
- $fd\_maximize(But, X)$

Bref ...

- Un programme de résolution de contraintes s'écrit en 3 parties :
  - définir les contraintes des variables,
  - décrire la contrainte,
  - résoudre
- Exemple :
  - $probleme([X, Y, Z]) :$
    $-fd\_domain(X, 0, 5), fd\_domain([Y, Z], 3, 7), X + Y\# <$
    $2 * Z, fd\_labeling([X, Y, Z]).$