

Programmation Logique et Par Contraintes Avancée

Cours 9 – Symétries et Contraintes Redondantes

Ralf Treinen

Université Paris Cité
UFR Informatique
Institut de Recherche en Informatique Fondamentale



`treinen@irif.fr`

4 mars 2025

Symétries

Qu'est-ce que c'est une symétrie dans un problème de résolution de contrainte c par rapport à un domaine initial D ?

- ▶ Soit A l'ensemble des affectation des variables en c qui sont cohérentes avec D .
- ▶ Soit S l'ensemble des solutions de c par rapport à D :
 $S \subseteq A$.
- ▶ Il y a une fonction de *normalisation* $Norm : A \rightarrow A$ telle que :
 - ▶ $Norm$ est idempotent : $Norm(Norm(x)) = Norm(x)$
 - ▶ On peut facilement calculer $Norm^{-1}(x)$
 - ▶ $x \in S \Leftrightarrow Norm(x) \in S$
- ▶ On dit que x est *normal* (par rapport à N) quand $x = N(x)$.

L'intérêt d'identifier des symétries

- ▶ Il suffit de chercher des solutions σ qui sont normales
- ▶ On peut essayer d'ajouter une contrainte qui exprime le fait qu'une solution est normale (élimination de symétries), mais ce n'est pas toujours possible (voir les exemples plus tard).
- ▶ On peut, si on le souhaite, obtenir facilement l'ensemble de toutes les solutions puisqu'on peut évaluer facilement $Norm^{-1}$.

Exemple de symétrie : N dames

- ▶ Problème des n dames : Variables X_1, \dots, X_n :
- ▶ Chaque variable correspond à une ligne, sa valeur est la colonne dans laquelle se trouve la dame de cette ligne.
- ▶ On peut supposer que la dame de la ligne 1 se trouve dans la partie gauche du damier.
- ▶ Fonction de normalisation

$$N(\sigma) = \begin{cases} \sigma & \text{si } \sigma(X_1) \leq \lceil n/2 \rceil \\ \sigma' : \sigma'(X) = n - \sigma(X) & \text{sinon} \end{cases}$$

- ▶ Contrainte supplémentaire : $X_1 \leq \lceil n/2 \rceil$.

Exemple de symétrie : coloration d'une carte

- ▶ Coloration d'une carte à n pays : variables C_1, \dots, C_n .
- ▶ Chaque variable correspond à un pays, sa valeur est la couleur de ce pays.
- ▶ On peut fixer la couleur du premier pays, par exemple couleur 1.
- ▶ Fonction de normalisation :

$$N(\sigma) = \begin{cases} \sigma & \text{si } \sigma(C_1) = 1 \\ \sigma \text{ avec 1 et } \sigma(C_1) \text{ échangés} & \text{sinon} \end{cases}$$

- ▶ Contrainte supplémentaire : $C_1 = 1$.

Éliminer plus de symétries ?

- ▶ On pourrait même imposer que :
 - ▶ Si $\sigma(C_i) \notin \{\sigma(C_j) \mid j < i\}$
 - ▶ Alors $\sigma(C_i) > \sigma(C_j)$ pour tout $i > j$
- ▶ Cela éliminerait toutes les symétries dues aux permutations des couleurs
- ▶ Problème : on ne sais pas réaliser cette contrainte.
- ▶ Il existent seulement des solutions imparfaites, par exemple par identification des parties connexes du graphe.

Contraintes redondantes

- ▶ Une contrainte c' est *redondante* par rapport à une contrainte c si $c \models c'$.
- ▶ Le propagateur d'une contrainte redondante c' peut permettre de faire des réductions de domaines que les propagateurs de c ne peuvent pas faire.
- ▶ Il peut être intéressant d'ajouter des propagateurs pour des contraintes redondantes.

Exemple : des carrés magiques

Un carré magique de taille n est

- ▶ une matrice $n \times n$ avec des valeurs entre 1 et n^2
- ▶ toutes les cases ont une valeur différente
- ▶ les sommes des lignes, des colonnes, et des deux diagonales sont égales
- ▶ Exemple pour $n = 3$:

2	7	6
9	5	1
4	3	8

Exemples (magic1.oz) I

```

declare
fun {MagicSquare N}
    NN = N*N
    L1N = {List.number 1 N 1} % [1 2 3 ... N]
in
    proc {$ Square}
        Sum = {FD.decl}
        fun {Field I J} % domaine en ligne I colonne J
            Square.((I-1)*N + J)
        end
        proc {Assert F}
            % {F 1} + {F 2} + ... + {F N} =: Sum
            {FD.sum {Map L1N F} '=' Sum}
        end
    in
        {FD.tuple square NN 1#NN Square}
        {FD.distinct Square}
    
```

Exemples (magic1.oz) II

```

%% Diagonals
{Assert fun {$ I} {Field I I} end}
{Assert fun {$ I} {Field I N+1-I} end}
%% Rows
for R in 1..N do
    {Assert fun {$ J} {Field R J} end}
end
%% Columns
for C in 1..N do
    {Assert fun {$ I} {Field I C} end}
end
{FD.distribute split Square}
end
end

declare
proc {MyExec P}

```

Exemples (magic1.oz) III

```
    Start End
in
    Start={OS.time}
    {Browse {SearchAll P}}
    End={OS.time}
    {Show (End-Start)}
end

{MyExec {MagicSquare 4}}
```

Amélioration : éliminer des symétries

- ▶ La version naïve prend 31 secondes (n=4)
- ▶ Normalisation : on peut tourner tout carré tel que la valeur dans la case (1,1) est minimale parmi les quatre coins.
- ▶ On peut en plus tourner le carré tel que la case (N,1) est \leq la case (1,N).
- ▶ Donc contraintes supplémentaire :

$$X_{1,1} < X_{n,1}, X_{n,1} < X_{1,n}, X_{1,1} < X_{n,n}$$

- ▶ Temps d'exécution avec ces propagateurs supplémentaires (n=4) : 8 secondes

L'exemple modifié (magic2.oz) I

Ajouter les lignes :

```
%% Eliminate symmetries  
{Field 1 1} <: {Field N N}  
{Field N 1} <: {Field 1 N}  
{Field 1 1} <: {Field N 1}
```

Ajouter un propagateur pour une contraintes redondante

- ▶ La somme de toutes les cases est

$$1 + 2 + \dots + n^2 = \frac{n^2 * (n^2 + 1)}{2}$$

- ▶ La somme de toutes les cases est égale à n fois la somme d'une ligne (colonne, diagonale)
- ▶ Contrainte redondante : $\frac{n^2 * (n^2 + 1)}{2} = n * sum$
- ▶ Temps d'exécution avec ce propagateur supplémentaire (n=4) : < 1 seconde.

L'exemple encore modifié (magic3.oz) I

Ajouter les lignes :

```
%% Redundant: sum of all fields = (number rows) * Sum  
NN*(NN+1) div 2 =: N*Sum
```

Autres contraintes redondantes ?

- ▶ On avait ajouté :

$$X_{1,1} < X_{n,1}, X_{n,1} < X_{1,n}, X_{1,1} < X_{n,n}$$

- ▶ Contrainte impliquée (donc redondante) :

$$X_{1,1} < X_{1,n}$$

- ▶ Est-ce qu'on gagne quelque chose si on ajoute le propagateur pour cette contrainte redondante ?