Rapport

Projet INF402 - Introduction à la logique

Groupe INF-1

Auteurs : (Groupe N)
Ibrahim ALSABR
Jean BOU SERHAL

Référent :
Astor BIZARD

N reines

6 mai 2021

1. Choix du sujet et présentation du problème en français :

Le problème que nous avons choisi de résoudre est celui des **N reines**, à l'aide d'un SAT-solveur.

Comment placer N reines sur un échiquier de taille $N * N$ de façon à ce qu'aucune reine ne puisse attaquer l'autre ?

Les mouvements des reines étants droits, verticalement, horizontalement et diagonalement ; ce qui implique les contraintes sur clauses qu'on précisera ci-après.

Voici un exemple de solution du problème des 4 reines : (normalement résolu en suivant un algorithme de backtracking et qui peut présenter plusieurs solutions suite à N^Nconfigurations)

	1	2	3	4
1		R		
2				R
3	R			
4			R	

Les contraintes sur les clauses viennent des possibilités de mouvement de la reine sans qu'elle n'attaque une autre reine. Il n'y aura alors qu'obligatoirement une et une seule (1) reine par ligne, obligatoirement une et une seule (1) reine par colonne, et au plus une (0 ou 1) reine par diagonale ou toute autre direction parallèle aux diagonales.

2. Modélisation en logique du premier ordre :

Le prédicat $R(l,c)$ représente le fait que la case présente à la ligne l et à la colonne c est occupée par une reine. l et c appartenant à l'intervalle des entiers entre 1 et N ($l,c \in [1;N]$).

$R(l,c)$ est $vraie$ si une reine occupe la case ; elle est $fausse$ dans le cas contraire.

Les contraintes du problème se modélisent comme suit :

- Il n'y a qu'exactement une reine par ligne : (C1)
  - Chaque reine est sur une ligne : $\forall l, \exists c, R(l,c)$
  - Il y a au plus une reine par ligne : $\forall l, \forall m, l \neq m \Longrightarrow \forall c, \overline{R(l,c)} \vee \overline{R(m,c)}$ (Le nombre de lignes étant égal au nombre de reines, il n'est pas nécessaire de formaliser une contrainte « au moins une reine par ligne ».)
- Il n'y a qu'exactement une reine par colonne : (C2)
  - Chaque reine est sur une colonne : $\forall c, \exists l, R(l,c)$
  - Il y a au plus une reine par colonne : $\forall c, \forall d, c \neq d \Longrightarrow \forall l, \overline{R(l,c)} \vee \overline{R(l,d)}$ (Le nombre de colonnes étant égal au nombre de reines, il n'est pas nécessaire de formaliser une contrainte « au moins une reine par colonne ».)
- Il y a au plus une reine par diagonale : (C3) ? en revanche il y a la modélisation en CNF.

3. Modélisation en forme normale conjonctive :

La variable booléenne $r_{i,j}$ représente le fait qu'une reine remplie la case de la ligne i et de la colonne j. Par exemple, pour le problème des 4 reines, avec $i \in \{1,2,3,4\}$ et $j \in \{1,2,3,4\}$, nous devons donner au SAT-solveur les contraintes suivantes :

$(r_{1,1} \vee r_{1,2} \vee r_{1,3} \vee r_{1,4})$ C1
$\wedge (r_{2,1} \vee r_{2,2} \vee r_{2,3} \vee r_{2,4})$
$\wedge (r_{3,1} \vee r_{3,2} \vee r_{3,3} \vee r_{3,4})$
$\wedge (r_{4,1} \vee r_{4,2} \vee r_{4,3} \vee r_{4,4})$
$\wedge (\overline{r_{1,1}} \vee \overline{r_{1,2}}) \wedge (\overline{r_{1,1}} \vee \overline{r_{1,3}}) \wedge (\overline{r_{1,1}} \vee \overline{r_{1,4}}) \wedge (\overline{r_{1,2}} \vee \overline{r_{1,3}}) \wedge (\overline{r_{1,2}} \vee \overline{r_{1,4}}) \wedge (\overline{r_{1,3}} \vee \overline{r_{1,4}})$
$\wedge (\overline{r_{2,1}} \vee \overline{r_{2,2}}) \wedge (\overline{r_{2,1}} \vee \overline{r_{2,3}}) \wedge (\overline{r_{2,1}} \vee \overline{r_{2,4}}) \wedge (\overline{r_{2,2}} \vee \overline{r_{2,3}}) \wedge (\overline{r_{2,2}} \vee \overline{r_{2,4}}) \wedge (\overline{r_{2,3}} \vee \overline{r_{2,4}})$
$\wedge (\overline{r_{3,1}} \vee \overline{r_{3,2}}) \wedge (\overline{r_{3,1}} \vee \overline{r_{3,3}}) \wedge (\overline{r_{3,1}} \vee \overline{r_{3,4}}) \wedge (\overline{r_{3,2}} \vee \overline{r_{3,3}}) \wedge (\overline{r_{3,2}} \vee \overline{r_{3,4}}) \wedge (\overline{r_{3,3}} \vee \overline{r_{3,4}})$
$\wedge (\overline{r_{4,1}} \vee \overline{r_{4,2}}) \wedge (\overline{r_{4,1}} \vee \overline{r_{4,3}}) \wedge (\overline{r_{4,1}} \vee \overline{r_{4,4}}) \wedge (\overline{r_{4,2}} \vee \overline{r_{4,3}}) \wedge (\overline{r_{4,2}} \vee \overline{r_{4,4}}) \wedge (\overline{r_{4,3}} \vee \overline{r_{4,4}})$

$\wedge (r_{1,1} \vee r_{2,1} \vee r_{3,1} \vee r_{4,1})$ C2
$\wedge (r_{1,2} \vee r_{2,2} \vee r_{3,2} \vee r_{4,2})$
$\wedge (r_{1,3} \vee r_{2,3} \vee r_{3,3} \vee r_{4,3})$
$\wedge (r_{1,4} \vee r_{2,4} \vee r_{3,4} \vee r_{4,4})$
$\wedge (\overline{r_{1,1}} \vee \overline{r_{2,1}}) \wedge (\overline{r_{1,1}} \vee \overline{r_{3,1}}) \wedge (\overline{r_{1,1}} \vee \overline{r_{4,1}}) \wedge (\overline{r_{2,1}} \vee \overline{r_{3,1}}) \wedge (\overline{r_{2,1}} \vee \overline{r_{4,1}}) \wedge (\overline{r_{3,1}} \vee \overline{r_{4,1}})$
$\wedge (\overline{r_{1,2}} \vee \overline{r_{2,2}}) \wedge (\overline{r_{1,2}} \vee \overline{r_{3,2}}) \wedge (\overline{r_{1,2}} \vee \overline{r_{4,2}}) \wedge (\overline{r_{2,2}} \vee \overline{r_{3,2}}) \wedge (\overline{r_{2,2}} \vee \overline{r_{4,2}}) \wedge (\overline{r_{3,2}} \vee \overline{r_{4,2}})$
$\wedge (\overline{r_{1,3}} \vee \overline{r_{2,3}}) \wedge (\overline{r_{1,3}} \vee \overline{r_{3,3}}) \wedge (\overline{r_{1,3}} \vee \overline{r_{4,3}}) \wedge (\overline{r_{2,3}} \vee \overline{r_{3,3}}) \wedge (\overline{r_{2,3}} \vee \overline{r_{4,3}}) \wedge (\overline{r_{3,3}} \vee \overline{r_{4,3}})$
$\wedge (\overline{r_{1,4}} \vee \overline{r_{2,4}}) \wedge (\overline{r_{1,4}} \vee \overline{r_{3,4}}) \wedge (\overline{r_{1,4}} \vee \overline{r_{4,4}}) \wedge (\overline{r_{2,4}} \vee \overline{r_{3,4}}) \wedge (\overline{r_{2,4}} \vee \overline{r_{4,4}}) \wedge (\overline{r_{3,4}} \vee \overline{r_{4,4}})$

$\wedge (\overline{r_{3,1}} \vee \overline{r_{4,2}})$ C3
$\wedge (\overline{r_{1,1}} \vee \overline{r_{2,2}}) \wedge (\overline{r_{1,1}} \vee \overline{r_{3,3}}) \wedge (\overline{r_{1,1}} \vee \overline{r_{4,4}}) \wedge (\overline{r_{2,2}} \vee \overline{r_{3,3}}) \wedge (\overline{r_{2,2}} \vee \overline{r_{4,4}}) \wedge (\overline{r_{3,3}} \vee \overline{r_{4,4}})$

$\wedge (\overline{r_{1,3}} \vee \overline{r_{2,4}})$
$\wedge (\overline{r_{1,2}} \vee \overline{r_{2,3}}) \wedge (\overline{r_{1,2}} \vee \overline{r_{3,4}}) \wedge (\overline{r_{2,3}} \vee \overline{r_{3,4}})$
$\wedge (\overline{r_{2,1}} \vee \overline{r_{3,2}}) \wedge (\overline{r_{2,1}} \vee \overline{r_{4,3}}) \wedge (\overline{r_{3,2}} \vee \overline{r_{4,3}})$

$\wedge (\overline{r_{2,1}} \vee \overline{r_{1,2}})$
$\wedge (\overline{r_{4,1}} \vee \overline{r_{3,2}}) \wedge (\overline{r_{4,1}} \vee \overline{r_{2,3}}) \wedge (\overline{r_{4,1}} \vee \overline{r_{1,4}}) \wedge (\overline{r_{3,2}} \vee \overline{r_{2,3}}) \wedge (\overline{r_{3,2}} \vee \overline{r_{1,4}}) \wedge (\overline{r_{2,3}} \vee \overline{r_{1,4}})$
$\wedge (\overline{r_{4,3}} \vee \overline{r_{3,4}})$

$\wedge (\overline{r_{3,1}} \vee \overline{r_{2,2}}) \wedge (\overline{r_{3,1}} \vee \overline{r_{1,3}}) \wedge (\overline{r_{2,2}} \vee \overline{r_{1,3}})$
$\wedge (\overline{r_{4,2}} \vee \overline{r_{3,3}}) \wedge (\overline{r_{4,2}} \vee \overline{r_{2,4}}) \wedge (\overline{r_{3,3}} \vee \overline{r_{2,4}})$

4. Le programme :

Le programme Générateur_DIMACS en langage Python que nous avons écrit, nous permet en pré-saisissant la valeur N, d'écrire le fichier DIMACS requis pour le faire passer au SAT-solveur. Il suffit juste de préciser dans la configuration la valeur de l'argument.

Le programme est composé de

- 4 fonctions, dont :
  - 3 fonctions qui nous permettent d'exprimer les différentes contraintes sur les clauses qui définissent les règles du jeu (une et une seule reine par ligne ou par colonne, c-à-d au moins une ET au plus une ; au moins une reine par diagonale)
  - 1 fonction qui permet de positionner la variable sur l'échiquier.
- Une séquence de boucles qui nous permet d'écrire dans un fichier ouvert les clauses conformément aux contraintes, en se déplaçant par ligne, par colonne, puis par diagonale ou par parallèle à cette dernière.

Le programme Lecteur_TraceSAT permet d'afficher la solution d'une manière compréhensible sous forme d'un échiquier, à partir de la trace récupérée auprès du SAT-solveur.

5. Exemples :

Pour N = 4, le programme Générateur_DIMACS génère un fichier .cnf « 4-reines.cnf ». Ce fichier donne un résultat satisfaisable « SATISFIABLE » dans le SAT-solveur et présente à la ligne v, une solution.

Alors que pour N = 6, « UNSATISFIABLE », donc pas de solution.