Correction TD IA - 2026

Équipe Pédagogique IA

1 TD5 : Satisfaction de Contraintes (CSP) et Logique (SAT)

Exercice 5 : Modélisation SAT du Sudoku (Exemple 4x4)

Énoncé : Modélisez le problème pour une grille 4×4 (pour simplifier les calculs). Analysez le nombre de variables et de formules.

1	2		
		2	1
2	4		
		4	2

FIGURE 1 – Exemple d'instance Sudoku 4x4 à résoudre

Proposition de Corrigé

1. Définition des Variables

On définit les variables propositionnelles $x_{i,j,k}$ telles que :

$$x_{i,j,k} = \text{Vrai} \iff \text{La case (ligne } i, \text{colonne } j\text{) contient la valeur } k$$

Pour un Sudoku 4x4 : $i, j, k \in \{1..4\}$.

Calcul du nombre de variables :

$$4 \text{ (lignes)} \times 4 \text{ (colonnes)} \times 4 \text{ (valeurs)} = \mathbf{64} \text{ variables.}$$

2. Contraintes (Formules)

A. Contrainte de représentation de l'état initial Si la grille contient un 2 en position (2,3), on ajoute la clause unitaire : $(x_{2,3,2})$.

B. Contrainte : "Une case contient au moins une valeur" Pour chaque case (i, j), il faut qu'au moins l'une des valeurs soit vraie :

$$\forall i, j : (x_{i,j,1} \lor x_{i,j,2} \lor x_{i,j,3} \lor x_{i,j,4})$$

Pour la case de la première ligne ($i = 1$) et première colonne ($j = 1$), cela donne :

$$(x_{1,1,1} \lor x_{1,1,2} \lor x_{1,1,3} \lor x_{1,1,4})$$

$\rightarrow$ 1 clause par case $\times$ 16 cases = **16 formules**.

C. Contrainte : "Une case contient au plus une valeur" C'est ici qu'on peut modéliser de deux façons.

— **Solution 1 : Approche Logique (Implication)** On dit : "Si la case vaut k, alors elle ne peut pas valoir les autres".
Exemple pour la case (1,1) et la valeur 1 :

$$x_{1,1,1} \implies \neg(x_{1,1,2} \vee x_{1,1,3} \vee x_{1,1,4})$$

$\rightarrow$ 4 formules par case $\times$ 16 cases = **64 formules**.
Total B + C (Solution 1) : $16 + 64 = 80$ formules.

— **Solution 2 : Approche CNF (Paires exclusives - XOR)** Les solveurs préfèrent les clauses simples. On interdit toutes les paires de valeurs pour une même case.
Pour une case donnée, on interdit :

  — $(\neg x_{i,j,1} \vee \neg x_{i,j,2})$
  — $(\neg x_{i,j,1} \vee \neg x_{i,j,3})$ ... et ainsi de suite.

Il y a 6 paires possibles parmi 4 valeurs.
$\rightarrow$ 6 clauses par case $\times$ 16 cases = **96 formules**.
Total B + C (Solution 2) : $16 + 96 = 112$ formules (clauses).

D. Contraintes de validité (Lignes, Colonnes, Carrés) Pour s'assurer que la grille est valide, il suffit d'imposer que **chaque chiffre apparaisse au moins une fois** dans chaque groupe (ligne, colonne, carré).

Raisonnement logique : Puisqu'il y a 4 cases dans une ligne et 4 chiffres possibles, si chaque chiffre apparaît au moins une fois, alors ils apparaissent tous exactement une fois (principe des tiroirs), à condition que les contraintes sur les cases (B et C) soient respectées.

— **Pour les lignes :** Pour chaque ligne i et chaque valeur k, k doit apparaître au moins une fois.
Exemple (Ligne 1 doit contenir un 1) :

$$(x_{1,1,1} \vee x_{1,2,1} \vee x_{1,3,1} \vee x_{1,4,1})$$

$\rightarrow$ 4 lignes $\times$ 4 valeurs = **16 formules**.

— **Pour les colonnes :** Idem.
$\rightarrow$ 4 colonnes $\times$ 4 valeurs = **16 formules**.

— **Pour les carrés :** Idem.
$\rightarrow$ 4 carrés $\times$ 4 valeurs = **16 formules**.

Note : On n'a pas besoin d'ajouter les clauses d'unicité (AMO) pour les lignes/colonnes ici, car elles sont implicites par la combinaison des contraintes précédentes.

Résumé total (Solution CNF standard) :

$$112 \text{ (Cases)} + 48 \text{ (Validité)} + \text{Clauses unitaires (indices)} = \mathbf{160} \text{ clauses} + \text{indices}.$$

TD 6 : Comparaison SAT vs CSP

Critère	**Approche SAT (4x4)**	**Approche CSP (4x4)**
Variables	**64** variables booléennes ($x_{i,j,k}$)	**16** variables entières ($X_{i,j}$)
Domaines	$\{0, 1\}$ (Vrai/Faux)	$\{1, 2, 3, 4\}$
Contraintes	80 à 112 formules logiques (Clauses CNF, bas niveau)	12 contraintes globales (`AllDifferent`, haut niveau)
Résolution	Solveurs très rapides	Plus compact à modéliser

Conclusion : Le CSP offre une modélisation beaucoup plus compacte ("Haut niveau") grâce aux variables entières, tandis que SAT ("Bas niveau") explose en nombre de variables mais bénéficie de solveurs extrêmement optimisés.