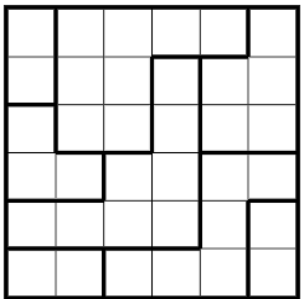
COMPTE RENDU 1

NORI-NORI

DUDEK Milan , RAVELONAHINA Miranto, ROUALY Mehdi , MBUY'IBUTSI Mathys

Explication du Nori-Nori:

On joue au Nori-Nori dans une grille de taille NxN (ici 6x6) comme suit où on coloriera des cases:

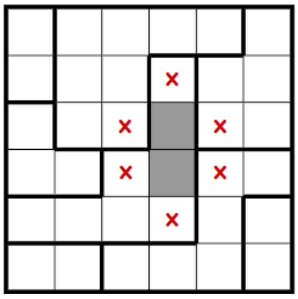


Le jeu comporte trois règles :

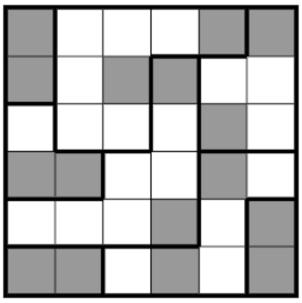
- Les cases doivent être coloriées de manière à faire des dominos
- Deux dominos ne peuvent pas être adjacents (ils peuvent l'être en diagonale)
- Chaques zones (délimitées en noir) doivent contenir exactement 2 cases coloriées

Nous décidons de rassembler les deux premières règle comme suit:

- Une case coloriée doit avoir exactement 1 case voisine coloriée (ex ci dessous)



Une solution de cette grille exemple est :



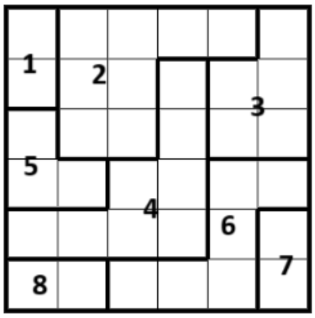
Modélisation au premier ordre :

On pose le prédicat suivant

- C(i,j) -> vrai si la case à la position i,j est coloriée

On définit l'ensemble P comme le produit cartésien NxN, dimension de la grille. La case(1,1) sera l'origine du repère et située en haut à gauche.

On numérote les zones x d'une grille de 1 à n, n étant le nombre de zones (ex ci dessous)



On définit enfin l'ensemble Y(x) comme le sous ensemble de P contenant les cases de la zone x

Ces ensembles nous permettent de préciser la nature des variables utilisées par la suite.

Modélisation des deux premières règles :

$\forall$i,j $\in$ P, C(i,j) $\Rightarrow$ [C(i-1,j) $\oplus$ C(i+1,j) $\oplus$ C(i,j-1) $\oplus$ C(i,j+1)]

Modélisation de la troisième règle :

$\forall$x $\in$ ⟦1,n⟧, $\exists$ (i;j),(k;l) $\in$ Y(x), [i,j ≠ k,l] ^ C(i,j) ^ C(k,l) ^ [$\forall$m,n $\in$ Y(x), m,n ≠ i,j ^ m,n ≠ k,l => !C(m,n)]

Modélisation sous forme FNC :

La variable booléenne X(i,j) représente le fait que la case à la position i,j soit colorée ou non.

Modélisation deux premières règles :

(désolé pour la lisibilité, on a pas réussi à faire mieux)

$\bigwedge_{(i,j)\in P}$ [!X(i,j) v !X(i-1,j) v !X(i+1,j) v !X(i,j-1) v !X(i,j+1)] ^
[!X(i,j) v !X(i-1,j) v !X(i+1,j) v X(i,j-1) v X(i,j+1)] ^
[!X(i,j) v !X(i-1,j) v X(i+1,j) v !X(i,j-1) v X(i,j+1)] ^
[!X(i,j) v !X(i-1,j) v X(i+1,j) v X(i,j-1) v !X(i,j+1)] ^
[!X(i,j) v X(i-1,j) v !X(i+1,j) v !X(i,j-1) v X(i,j+1)] ^
[!X(i,j) v X(i-1,j) v !X(i+1,j) v X(i,j-1) v !X(i,j+1)] ^
[!X(i,j) v X(i-1,j) v X(i+1,j) v !X(i,j-1) v !X(i,j+1)] ^
[!X(i,j) v X(i-1,j) v X(i+1,j) v X(i,j-1) v X(i,j+1)]

Modélisation de la troisième règle :

Sans les connaissances et la maîtrise des techniques de la logique du 1er ordre, nous ne pouvons pas aller plus loin que cette formule pour la modélisations de cette règles en FNC :

∀x ∈ Z, ∃i,j,k,l ∈ Y(x),
[i,j ≠ k,l] ^ [C(i,j)] ^ [C(k,l)] ^ [∀m,n ∈ Y(x), (m,n = i, j V m,n = k,l V !C(m,n))]

__**Points critiques :**__

Nous avons utilisé l'application BDDC pour calculer la FNC de la première règle depuis la modélisation en premier ordre, avec la commande :

cnf (x => (a xor b xor c xor d));

Nous considérons x comme la case coloriée et a, b, c et d comme ses voisines.
Cette FNC s'étend à tous les couples (i,j) ∈ P, d'où l'utilisation de $\bigwedge_{(i,j)\in P}$

Pour la troisième formule, une autre idée serait de passer par la table de vérité et récupérer toutes les fausses lignes, donc de prendre la FND équivalente à la formule de non F, pour récupérer la FNC équivalente à F avec la négation de cette FND. Nous ne savons pas vraiment comment faire pour autant.