Projet Heyawake en Prolog

Kévin Martins Da Veiga [20009472] et Anri Kennel [20010664]

Table des matières

1 Présentation du Projet

1.1 Introduction et règles du jeu

L'Heyawake est un puzzle joué dans une grille rectangulaire séparée en plusieurs aires dont le but est de colorer des cases en noir et blanc en accord avec les règles :

- → Certaines aires contiennent un chiffre qui montre combien il y a de cases noires dans l'aire
  - → Aire avec un 0 ne doit pas avoir de cases noires
  - → Aire avec un 1 contient une case noire, on répète ses étapes autant de fois qu'il le faut
- → Toutes les cases blanches doivent être connectées en un seul groupe, pas de groupes isolés, elles doivent toutes être reliées
- → Deux cases noires ne peuvent être l'un à côté de l'autre horizontalement et verticalement (en diagonale, c'est possible)
  - → Ça veut dire que les cases noires doivent n'être qu'entourer de cases blanches
- → Une rangée de case blanche ne peut pas être alignée au-delà de 2 aires

1.2 Réalisation

- → Mettre toutes les aires avec un 0 en blanc
- → Mettre les aires avec 1 et que d'une case en noire
- → Mettre des cases noires là où une rangée de cases blanches dans plus de 2 aires peut se faire
- → Toujours mettre du blanc autour des cases noires
  - → En suivant cette étape, quand on met une case noire dans une aire avec un chiffre 1, colore tout le reste de l'aire en blanc
- → Toujours vérifié que les cases blanches ne sont pas piégées entre des cases noires
  - → Ça implique que dans ce schéma suivant avec B une case blanche, N une case noire et X une case vide :
    - → B X B → B B B
    - → N B N → N B N
    - → B N B → B N B
    - → Le X doit obligatoirement devenir blanc pour laisser une sortie à la case blanche qui se trouve entre les cases noires

Le programme se lance en appelant le prédicat ***run*** avec comme argument une liste de listes des aires.

```
run([[0,0,1,1,2], [2,0,2,2,-1], [3,0,5,1,-1],
     [0,2,0,4,-1], [1,2,1,3,0], [3,2,5,2,0],
     [1,4,1,4,1], [2,3,5,5,4], [0,5,1,5,-1]]).
% [X1, Y1, X2, Y2, N] avec N le nombre de case noire dans l'aire
```

Cet appel correspond au puzzle ci-dessous

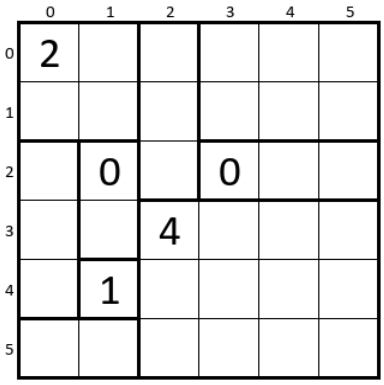

FIGURE 1 – Puzzle de difficulté facile d'Heyawake

2 Résolution d'une grille avec des aires

2.1 Ce que l'on a essayé

→ On vérifie les aires avec un 2 et de 4 cases puis on colorie la case 2 en noire et ses diagonales

  → On colore les cases adjacentes aux noires en blanc, on le fait autant de fois que nécessaire

→ On regarde quelles cases doivent être colorées en noire pour éviter de faire des lignes de 4 cases blanches

→ On connecte les cases blanches ensemble sans faire de lignes de plus de 3 cases dans des zones différentes

→ Maintenant on essaie de rajouter des cases noires :

  → On met une case noire pour éviter des groupes de 3 cases blanches

  → On colorie les cases adjacentes à la nouvelle case noire en blanc

→ On fait en sorte que toutes les cases blanches soient connectées en rajoutant des cases blanches si possible

→ On remplit les cases noires là où on peut les mettre, et on continue jusqu'à qu'il n'y est plus de cases sans couleurs.

3 Le programme

3.1 Le prédicat *aire* en détail

Le prédicat ***aire*** fonctionne comme l'image ci-contre. Il regarde chaque case de l'aire. Une fois arrivé à l'extrémité d'une aire, il descend d'une case s'il y une ligne en dessous puis il parcourt le chemin inverse avec le prédicat ***airereverse***.

Par exemple, sur la Figure 1, prenons l'aire en bas à droite :

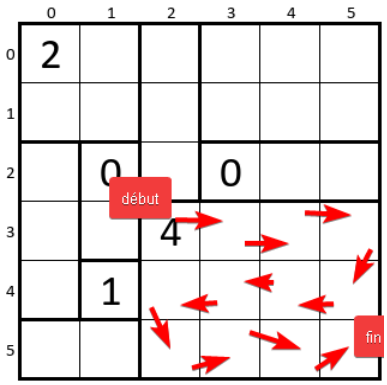

FIGURE 2 – Explication fonctionnement ***aire***

3.2 Liste des prédicats utilisés

→ `aire(X1, Y1, X2, Y2, E, T)` : Ce prédicat fait scanne l'aire renseignée et assigne une couleur via le prédicat ***case***.

→ `airereverse(X1, Y1, X2, Y2, 0, T)` : Ce prédicat est appelée par le prédicat ***aire*** et permet de vérifier tous les éléments d'une aire.

→ `case(X, Y, C)` : Ce prédicat renvoie la couleur d'une case du puzzle.

→ `run([[X1, Y1, X2, Y2, E] | Q])` : Ce prédicat lance la résolution du puzzle.

4 Annexe : le code

```
% zone noire
aire(X, Y, X, Y, 1) :- case(X, Y, 1).

% zone blanche
aire(X1, Y1, X2, Y2, 0, T) :- X1 \= X2, case(X1, Y1, 0), NX is X1
    + 1, aire(NX, Y1, X2, Y2, 0, T).
aire(X, Y1, X, Y2, 0, T) :- Y1 \= Y2, case(X, Y1, 0), NY is Y1 +
   1, airereverse(X, NY, X, Y2, 0, T).
```

```
7
8  % zone : E = Etat, T = Taille
9  aire(X1, Y1, X2, Y2, E, T) :- E \= 0, write("\n"), write(X1),
       write(" "), write(Y1),
10                                  write(" "), write(X2), write("
                                        "), write(Y2),
11                                  write(" "), write(E), write(" "
                                        ), write(T).
12
13 airereverse(X1, Y1, X2, Y2, 0, T) :- case(X1, Y1, 0), NX is X1 -
       1, NX \= T,
14                                  airereverse(NX, Y1, X2, Y2, 0, T
                                        );
15                                  NX = T, aire(NX, Y1, X2, Y2, 0,
                                        T).
16
17 % cellule
18 case(X, Y, C) :- write("\n"), write("Coordonnées: X="), write(X),
19                 write(", Y="), write(Y),
20                 write(" | Couleur: "), write(C).
21
22 run([]) :- write('0: case blanc, 1: case noire').
23 run([[X1, Y1, X2, Y2, E] | Q]) :- T is X2 - X1, aire(X1, Y1, X2,
       Y2, E, T), run(Q).
24
25 /*
26 run([[0,0,1,1,2],
27      [2,0,2,2,-1],
28      [3,0,5,1,-1],
29      [0,2,0,4,-1],
30      [1,2,1,3,0],
31      [3,2,5,2,0],
32      [1,4,1,4,1],
33      [2,3,5,5,4],
34      [0,5,1,5,-1]]).
35 */
```