Génération du labyrinthe

CARTAYA Agustin

Restrictions:

- Taille du labyrinthe = R * C (R = nombre de lignes ; C = nombre de colommnes)
- Les cases 0;0 et R-1;C-1 doivent être blanches
- Nombre de cases blanches proche ou égal à k (défini par l'utilisateur)
- Tous les cases blanches doivent être connectés
- les cases noires doivent être placés au hasard

1. **vérification que la création du labyrinthe avec les données données est possible, cas ou n'est pas possible :**

   Cases blanches insuffisants:
   soit un labyrinthe de taille R * C, le nombre minimum de cases voisines connexes qui connecte la case 0;0 et la R-1;C-1 est égal à {min = WR + C - 1} (voir «Distance de Manhattan») donc, si k < min. il faut augmenter le nombre de cases blanches k = R + C + 2 (on ajoute 2 pour des raisons d'efficacité lors de la création de très grands labyrinthes)

   Cases blanches en excès:
   Le cas contraire se produit lorsqu'on veut créer un labyrinthe avec plus de cases blanches qu'on n'en a pas c-a-d k > R * C dans ce cas les carrés blancs doivent être réduits et dans notre cas ils seront réduits de moitié, c'est-à-dire K = R * C / 2

2. **Génération d'un labyrinthe de n cases noires positionnés aléatoirement.**

   Pour la génération du tableau aléatoire de n cases noires (n = R * C - k), on a parcouru une ou plusieurs fois des casses dans des positions specuaux qui on a palle cases guides:

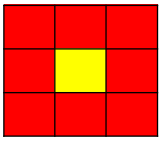
   I. Les cases noires guides (gi) sont celles où les numéros de ligne et de colonne sont impairs. Ces cases peuvent être cochées ou non

	0	1	2	3	4	5
0						
1		g0		g1		g2
2						
3		g3		g4		g5
4						
5		g6		g7		g8

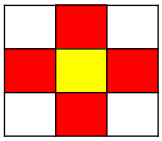
   II. tous les voisins valides de chaque case guide ont été recherchés et l'un d'eux est choisi au hasard pour le marquer s'il remplit les conditions suivantes:
   - N'est pas déjà marqué
   - N'est pas la case 0;0 ou R-1;C-1

   ✗ voisins valides: ce sont ceux qui entourent la boîte courante et ne sont pas marqués

   ✗ selon le niveau d'hasard choisi pour la génération du labyrinthe, la case guide est marque ou pas, et les voisins sont de type 1 ou 2

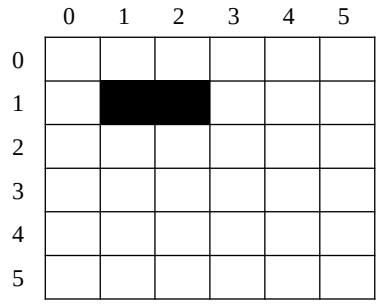


Voisins Type 1 de la case jaune

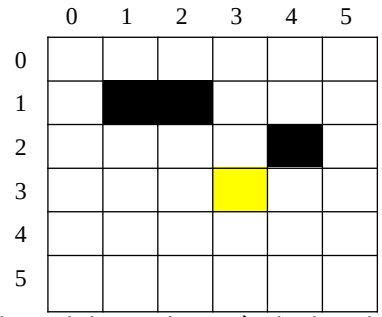


Voisins Type 2 de la case jaune

Processus de marquage



Marquer la case 1;1 et son voisin de droite



Marquer le voisin en haut à droite de la case 3;3 mais pas la case 3;3

les cases guides sont traitées dans un ordre aléatoire à l'aide d'un tableau de permutation comme expliqué ci-dessous:

- Le nombre de cases guides (ng) du tableau est: ng = [R/2] * [C/2]
  [x] = fonction ceil
- Un tableau de permutation (perm) est créé contenant les “ng” premiers entiers, exemple: ng = 9 → perm = 5, 7, 3, 6, 1, 2, 0, 8, 4
- A chaque tour est traité la case g(perm[i]) 0 <= i < ng
- pour marquer cette case dans le tableau original la conversion est faite par
  - a = 2 * perm[i] + 1
  - id = 2 * getLigne(a) + (rand() % 2) * 1) * C + a % C

Ce processus est répété jusqu'à qu’on obtient le nombre de cases noires souhaité,

3. Vérification de la connexité

Après avoir généré le labyrinthe de n cases noires, la connexité entre toutes ses cases est vérifiée (méthode de vérification expliquée dans la présentation du projet):

- Il existe des cases pas connexes
  Un autre labyrinthe est créé en suivant les étapes précédentes et il est vérifié si ce nouveau labyrinthe est bien connexe, cette étape est répété jusqu’à trouver un labyrinthe connexe.

  * si le nombre d’essais dépasse le nombre maximum d’essais (calculé à partir du nombre de cases dans la tableau), le k est augmenté en 1 a chaque 100 labyrinthes crées. Cela est fait pour augmenter la possibilité d'obtenir un labyrinthe connexe.

- Toutes les cases sont connexe:
  Le labyrinthe est déjà créé, il et toutes ses spécifications ainsi que la labyrinthe sont montrés à l'utilisateur

Méthodes utilisées

genLaby(int k)
Création du labyrinthe où k est le nombre de cases blanches. Il faut noter que la taille du labyrinthe est préalablement fixée par les variables NB_COLONNES, NB_LIGNES. Dans cette méthode, on vérifie que k est dans les plages autorisées et on procède à l'appel de la méthode genLabyInternal (…) autant de fois que nécessaire pour obtenir un labyrinthe connexe, s'il est nécessaire, k est progressivement augmenté.

genLabyInternal(int noirs, int alea)
Génération d'un labyrinthe avec «n» carrés noirs et avec un degré d'aléa entre 0 et 2 donné par la variable «alea», dans cette méthode on fait appel à d'autres méthodes pour nettoyer le labyrinthe, créer la permutation des cases guides, obtenir les voisins, entre autres.

nettoyer(int val)
mettre tous les carrés du labyrinthe avec une valeur de "val"

genPerm(int t[], int n)
créer une permutation des "n" premiers entiers et les stocker dans le tableau "t"

voisinsPointer(int id, char **t)
en prenant comme référence l'index i dans le tableau contenant le labyrinthe, un tableau de pointeurs est créé avec tous les voisins de i (pointant vers le tableau d'origine) et stocké dans "t"

Résultats

```
--------- Creation du labyranthe 6 x 10 --
labyranthe cree apres 4331 essais
   ...          Attendu         Reel
Noires:         30              30
Blanches:       30              30
Total:          60              60
------------------------------------------

   -----x---x
   -xxx---xxx
   ---x-x-xxx
   -xxxxx---x
   -xxxxx-xxx
   -xxx------
```

```
--------- Creation du labyranthe 6 x 10 -----
labyranthe cree apres 8716 essais
   ...          Attendu         Reel
Noires:         35              35
Blanches:       25              25
Total:          60              60
------------------------------------------

   -----xxxxx
   -xxxxxxxxx
   -xxx-xxxxx
   ---x-xxxxx
   x---------
   x-xx-xxx--
```

```
--------- Creation du labyranthe 10 x 10 -------
labyranthe cree apres 45120 essais
   ...          Attendu         Reel
Noires:         50              50
Blanches:       50              50
Total:          100             100
------------------------------------------------

   -xx-xxxx-x
   ---------x
   x-xx-x-x--
   ----xx-xxx
   xx-xxxxx-x
   xx---xx--x
   -xxx-xxx-x
   ---x-xxx-x
   -xxx-xxx-x
   ----------
```

```
--------- Creation du labyranthe 15 x 15 -
labyranthe cree apres 92 essais
   ...          Attendu         Reel
Noires:         75              75
Blanches:       150             150
Total:          225             225
------------------------------------------

   -----x---x--xx-
   -x-x-x---x-----
   -x-x---x----xxx
   -xxxxxx--xx-x-
   ---xxx-x-xxx---
   ---------------
   xxx------x--x--
   ----xxx--x--x-x
   -xx-xx--x--xxxx
   ----xx---xx----
   xx-xxx--x--x--x
   -x-x----x--xx-x
   -xxx-----------
   ---x--x-x------
   ---------------
```