Module : Algorithmique
Master Biologie Structurale et Bioinformatique

Devoir de TP libre
2018 - 2019

Jean-Michel Dischler

Cadre du problème

Un labyrinthe est un ensemble de chemins, avec un point de départ et un point d'arrivée (figure ci-dessous). Le joueur doit trouver un chemin reliant le départ et l'arrivée. Souvent on parle de point d'entrée et de sortie. Mais nous ne souhaitons pas nous limiter au cas où ces points seraient sur le bord du labyrinthe.

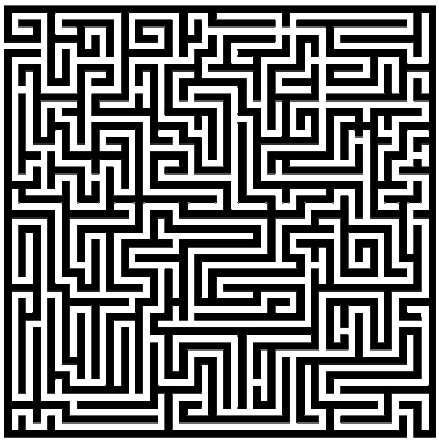

FIGURE 1 – Un exemple de labyrinthe

Certains labyrinthes sont dits parfaits. Dans un labyrinthe parfait, deux points quelconques peuvent toujours être reliés par un chemin et un seul. La figure 2 représente un labyrinthe parfait. Par contre, la figure 3 représente un labyrinthe imparfait. En effet, entre les points rouge et bleu, il y a deux chemins (en orange et en vert). Par ailleurs entre le point violet et le point rouge, il n'y a aucun chemin. Le point violet est sur un îlot.

FIGURE 2 – Un labyrinthe parfait

FIGURE 3 – Un labyrinthe imparfait

Création d'un labyrinthe

Dans cette première partie on se propose de coder en Java un générateur de labyrinthe.

Pour cela on se propose de coder le labyrinthe sous la forme d'une matrice de cellules carrées contenant une valeur booléenne : vrai signifie passage, faux signifie mur. On peut donc assimiler le labyrinthe à une matrice de pixels où chaque pixel est soit blanc soit noir (voir figure 1).

Java propose une classe « BitSet », similaire à la classe Vector où chaque élément de la liste est un booléen.

1. Définir une classe BitArray

Commencer par définir une nouvelle classe BitArray, utilisant la classe BitSet pour créer non pas des listes de booléens, mais des matrices de booléens, cette matrice ayant une taille X×Y.

On définira les opérations suivantes :

- Initialisation
- Affectation d'une valeur à un bit de coordonnées (i,j)
- Inversion d'une valeur de bit de coordonnées (i,j)
- Récupération de la valeur du bit de coordonnées (i,j)

D'autres opérations pourront être définies si nécessaire.

2. Génération d'un labyrinthe parfait

Pour représenter un labyrinthe, on définira un objet de notre classe BitArray nouvellement définie. Pour ce qui est de la génération d'un labyrinthe parfait, on on utilisera l'algorithme suivant:

a. On initialise le labyrinthe en plaçant des murs partout. L'objectif étant de « creuser » des passages.
b. On initialise une pile P de points courants.
c. On choisit un point de départ p, étant **le premier point courant**, que l'on empile sur P (vide initialement).
d. Puis on itère le processus suivant :
    1. A partir du point courant p regarder les 4 voisins (N, E, S, O)
    2. Constituer une liste Lv des voisins qui sont des murs et pour lesquels ce mur peut être remplacé par un passage libre, ***sans violer la règle du labyrinthe parfait***. A vous de déterminer quelles conditions doivent être vérifiées !
    3. Choisir un élément au hasard dans cette liste Lv, remplacer le mur choisi par un passage libre et placer la position courante p en ce nouveau point, empiler ce nouveau point sur P, puis recommencer à l'étape 1.
    4. Si la liste Lv est vide : càd qu'il n'est plus possible de « creuser » un passage dans cette direction, dépiler de P les positions précédentes jusqu'à tomber sur un point courant pour lequel des voisins « valides » existent (sa liste Lv n'est pas vide).

Afin d'éviter de gérer explicitement une pile P, on utilisera la récursivité pour écrire cet algorithme.

Utiliser l'algorithme précédent pour générer des labyrinthes de taille quelconque, complètement emmurés et dotés d'une entrée et d'une sortie sur le bord.

Créer une image de labyrinthe

Dans cette seconde partie on se propose de créer des fichiers d'images permettant d'afficher des labyrinthes.

Les formats « portable pixmaps »

Il existe de très nombreux formats d'images. Les uns rivalisent avec les autres pour représenter les images sur des fichiers les plus petits possibles avec le moins de perte d'information possible. Nous utiliserons ici le format pgm qui n'effectue aucune compression (et aucune perte d'information). Il fait partie d'une famille de formats (ppm, pgm, pbm) décrits plus en détail sur ce site https://fr.wikipedia.org/wiki/Portable_pixmap.
— le format pbm pour les images en noir et blanc ;
— le format pgm pour les images en niveaux de gris ;
— le format ppm pour les images en couleur.

L'avantage de ces formats est qu'il permet de représenter une image sous la forme d'un fichier texte. On peut donc le lire et le modifier *avec un simple éditeur de texte*.

Dans ce qui suit nous utilisons exclusivement le format pgm. Un fichier au format pgm :
— commence toujours par le mot P2 suivi d'un espace ou d'un retour à la ligne. Ces deux caractères sont appelés le magic number et permettent aux applications de lecture et d'édition d'image de reconnaître une image au format pgm
— doit ensuite comporter deux entiers représentant respectivement la largeur et la hauteur de l'image en pixels ;
— doit encore comporter un entier représentant l'intensité du blanc (généralement 255) ;
— tous les entiers qui suivent sont interprétés comme la valeur des pixels de l'image.

Par exemple le fichier pgm suivant :

```
P2
10 5
255
127 127 127 127 127 127 127 127 127 127 0 0 0 0 0 0 0 0 0 0
255 255 255 255 255
255 255 255 255 255 200 180 160 140 120 100 80 60 40 20
```

représente une image de 10 pixels de large et 5 pixels de haut. Le blanc est représenté par 255. Les trois premières lignes sont appelées l'entête du fichier image. Ensuite viennent une suite d'entiers représentant l'intensité de tous les pixels. La première valeur (ici 127) représente le pixel en haut à gauche de l'image. Ensuite viennent les pixels de la première ligne puis ceux de la deuxième ligne et ainsi de suite. L'image ci-dessus comporte une ligne de pixels (10

pixels) à 127, c'est à dire gris moyen. La deuxième ligne (10 pixels suivants) contint des valeurs à 0 (donc noir). Les 10 pixels suivants sont à 255 (donc blanc), etc.

Pour vous convaincre, lancer votre éditeur de texte préféré. Ouvrir un nouveau document et copier-coller le contenu ci-dessus dans ce fichier. L'enregistrer avec un nom *ayant une extension .pgm* (supposons que vous l'ayez appelée *premier.pgm*). Ca y est ! Vous avez créé très simplement un fichier image. Pour le voir, vous pouvez trouver le fichier dans un navigateur de fichier et double-cliquer dessus. L'image s'affichera.

Connaissant à présent le format de fichier pgm, écrire un programme permettant d'enregistrer un labyrinthe créé par l'algorithme précédent dans un fichier image.

Attention un pixel d'image ne correspond pas nécessairement à un mur. En effet, chaque mur ou passage correspondra à un carré de taille de N×N pixels dans l'image, N étant un paramètre de la fonction d'enregistrement. Un N grand (par exemple N=10) permettra de faire de « grandes » images même pour des labyrinthes très petits (comme ceux des figures 2 et 3).

Trouver un chemin dans un labyrinthe

Dans cette dernière partie on se propose de trouver le chemin d'un point de départ à un point d'arrivée. Les deux points sont choisis par l'utilisateurs (ils doivent être situés sur des passages libres). L'algorithme de recherche de chemin doit fournir la liste des directions à suivre pour aller du premier point vers le second, en indiquant à chaque fois s'il faut poursuivre vers le N, E, S ou O.

Par exemple, pour la figure 2, pour aller du point bleu vers le point rouge le programme indiquera la solution: O, N, E, S, E, N, O, N, E, N, O, N, O, S, O, N
Notez qu'il y a autant de lettres que de bifurcations.

Ecrire également un programme permettant de créer une image du chemin au sein du labyrinthe : pour le distinguer des passages libres en blanc, le chemin sera en gris moyen (valeur 127).

Remarques

Le TP libre est à rendre au plus tard pour le **lundi 3 décembre 2018**. Il sera envoyé sous forme électronique. Le fichier électronique contiendra :

- Les codes sources du programme Java.
- Une image pgm de labyrinthe généré, avec un chemin tracé en gris moyen.
- Un document word/rtf/pdf illustrant et documentant la réalisation (max. 5 pages).

Les projets sont à réaliser seuls ou en binômes.

La note est fonction de la quantité et qualité du travail* personnel *réalisé.