TP 3: sortir du labyrinthe

Judicaël Courant

2 novembre 2016

1 Présentation

Le but de cet exercice est de trouver la sortie d'un labyrinthe. Vous téléchargerez sur le site de la classe le fichier `labyrinthe.py` permettant de construire un labyrinthe et de le dessiner.

Pour l'essayer, vous pouvez exécuter :

```
p = 10
q = 20
t = construit_labyrinthe(p, q)
dessine_labyrinthe(t)
pl.show()
```

Chaque pièce du labyrinthe a des coordonnées entières dans $[\![0, p[\![\times [\![0, q[\![$. Vous trouverez dans le fichier `labyrinthe.py` une description de la façon dont le labyrinthe est représenté.

Le but de cet exercice est de parcourir le labyrinthe en utilisant la méthode du *parcours en profondeur*, utilisant une structure de pile. La pièce de coordonnées $(0, 0)$ est supposée être l'entrée du labyrinthe et celle de coordonnées $(p-1, q-1)$ est supposée être la sortie.

L'algorithme de parcours en profondeur d'un labyrinthe de taille $p \times q$ peut se décrire informellement comme suit :

1. On utilise une pile, destinée à contenir les coordonnées de certaines pièces du labyrinthe et contenant initialement uniquement les coordonnées de la pièce $(0, 0)$ du labyrinthe.
2. On dépile le sommet de la pile, obtenant ainsi les coordonnées (i, j) d'une certaine pièce du labyrinthe.
3. Si $(i, j) = (p - 1, q - 1)$, alors on sait qu'on peut aller de la pièce $(0, 0)$ à la pièce $(p - 1, q - 1)$ et on s'arrête. Sinon, on continue en passant au point suivant.
4. Pour chacune des pièces (i', j') qui ne sont pas séparées de (i, j) par un mur, on effectue les opérations suivantes. Si (i', j') a déjà été rencontrée, on ne fait rien. Sinon, on la met dans la pile.
5. On reprend à l'étape 2.

S'il y a bien un chemin de l'entrée à la sortie, la pile n'est jamais vide et l'algorithme termine.

2 Travail demandé

Le but est d'écrire une fonction `trouve_chemin(t)` prenant en entrée un labyrinthe `t` et trouvant un chemin reliant l'entrée à la sortie.

Cette fonction devra superposer, au dessin effectué par `dessine_labyrinthe(t)` le chemin trouvé.

Indication : une façon de trouver ce chemin est d'utiliser une méthode proche de celle du Petit Poucet : on garde une matrice `m` telle que pour toute pièce rencontrée (i, j), `m`$[i][j]$ donne les coordonnées de la pièce (i', j') dont on venait lorsqu'on a rencontré cette pièce. Il est alors facile de tracer son chemin en arrière à partir de la case $(p-1, q-1)$ pour revenir à la case $(0, 0)$.

