SAE_2.02 : EXPLORATION ALGORITHMIQUE

LE TOUR DU CAVALIER

SOMMAIRE

I - INTRODUCTION

Particularités

- but du projet
- chemin Hamiltonien
- cycle Hamiltonien

Difficultés

- DFS avec Backtracking

Ajouts

- Interface tkinter

II - Description claire du type d'algorithmes

POINTS COMMUNS

- choisir_case_aléatoire
- parcours_cavalier
- creer_interface

PREMIER ALGORITHME

- creer_graphe
- parcours
- tour
- itératif
- ne fonctionne pas

SECOND ALGORITHME

- trouver_chemin
- trouver_chemin_boucle
- récursif
- fonctionne

```python
def parcours_cavalier(taille):
    """
    Trouve un chemin pour un cavalier d'échecs qui passe par toutes les cases
    d'un plateau d'échecs de taille x taille sans jamais passer deux fois sur la même case.
    """
    graphe = creer_graphe(taille)

    choix1 = 0

    while(choix1 != 1 and choix1 != 2):
        choix1 = int(input("1 - choisir vous même la position de départ \n2 - Position de départ aléatoire \nVotre Choix : "))
        if (choix1 == 1):
            depart = int(input("Entrez le numéro de la case de départ : "))
        elif(choix1 == 2):
            depart = choisir_case_aleatoire(taille)
            print(depart)
        else:
            print("Erreur")
    plateau = [False] * taille * taille
    choix = 0
    while(choix != 1 and choix != 2):
        choix = int(input("1 - chemin Hamiltonien \n2 - cycle Hamiltonien \nVotre choix : "))
        if (choix == 1):
            chemin = trouver_chemin(taille, graphe, depart, plateau)
        elif(choix == 2):
            chemin = trouver_chemin_boucle(taille, graphe, depart, depart, plateau)
        else:
            print("Erreur")

    return chemin
```

```python
def trouver_chemin_boucle(taille, graphe, case_actuelle, depart, plateau):
    """
    Fonction récursive qui trouve un chemin pour un cavalier d'échecs qui passe par toutes les cases
    d'un plateau d'échecs de taille x taille sans jamais passer deux fois sur la même case.
    """
    plateau[case_actuelle] = True

    if all(plateau):
        # Toutes les cases ont été visitées, on a trouvé une solution
        if depart in graphe[case_actuelle]:
            return [case_actuelle, depart]
    voisins = graphe[case_actuelle]
    voisins = sorted(voisins, key=lambda v: len(graphe[v]))
    for voisin in voisins:
        if not plateau[voisin]:
            chemin = trouver_chemin_boucle(taille, graphe, voisin, depart, plateau.copy())
            if chemin is not None:
                return [case_actuelle] + chemin
    return None
```

III - Une comparaison argumentée

Explication des tests :

- performances
- types de graphes

Tests effectués

Echiquier de 8x8 avec cycle:

premier algorithme

```
Saisir la taille du plateau : 8
1 - choisir vous même la position de départ
2 - Position de départ aléatoire
Votre Choix : 2
57
1 - chemin Hamiltonien
2 - cycle Hamiltonien
Votre choix : 1
temps d'exécution :  0.0005130767822265625
deque([57, 51, 45, 39, 54, 60, 50, 44, 38, 55, 61, 46, 63, 53, 47,
, 22, 12, 6, 23, 29, 35, 41, 56, 26, 32, 17, 11, 5, 15, 21, 27, 33
```

second algorithme

```
Saisir la taille du plateau : 8
1 - choisir vous même la position de départ
2 - Position de départ aléatoire
Votre Choix : 1
Entrez le numéro de la case de départ : 52
1 - chemin Hamiltonien
2 - cycle Hamiltonien
Votre choix : 1
temps d'exécution :  0.01581287384033203
[52, 62, 47, 30, 15, 5, 11, 1, 16, 33, 48, 58, 41, 56, 50, 40, 57,
, 63, 46, 31, 14, 4, 19, 9, 24, 34, 44, 54, 39, 29, 35, 18, 3, 13,
```

Tests effectués

Echiquier de 8x8 sans cycle:

premier algorithme

```
Saisir la taille du plateau : 8
1 - choisir vous même la position de départ
2 - Position de départ aléatoire
Votre Choix : 1
Entrez le numéro de la case de départ : 57
1 - chemin Hamiltonien
2 - cycle Hamiltonien
Votre choix : 2
deque([40, 42])
temps d'exécution :  0.0009913444519042969
deque([57, 51, 45, 39, 54, 60, 50, 44, 38, 55, 61
, 22, 12, 6, 23, 29, 35, 41, 56, 26, 32, 17, 11,
```

second algorithme

```
Saisir la taille du plateau : 8
1 - choisir vous même la position de départ
2 - Position de départ aléatoire
Votre Choix : 2
52
1 - chemin Hamiltonien
2 - cycle Hamiltonien
Votre choix : 2
temps d'exécution :  0.017394304275512695
[52, 62, 47, 30, 15, 5, 11, 1, 16, 33, 48, 58,
, 63, 46, 31, 14, 4, 19, 9, 24, 34, 44, 54, 39
```

Résultats

- **Etude des résultats**
- **Comparaison des résultats**

IV - Amélioration / Solution

AMÉLIORATION

- Coins inferieur gauche et droit ne fonctionnent pas

SOLUTION

- solution non trouvée

Merci de nous avoir écouté