NSI · TERMINALE / COURS

Parcourir un graphe

Astrolab Cours · astrolabcours.fr

01 Le problème

Dans un réseau social, combien d'intermédiaires séparent deux personnes ? Dans un labyrinthe, existe-t-il une sortie ? Pour répondre, il faut visiter les sommets d'un graphe de façon organisée, sans en oublier et sans tourner en rond.

01 L'idée : la tache d'encre et l'explorateur

Une goutte d'encre sur un buvard s'étale en cercles : d'abord les voisins immédiats, puis les voisins des voisins. C'est le **parcours en largeur**. Un explorateur dans un labyrinthe, lui, fonce dans un couloir jusqu'au bout, puis revient au dernier carrefour : c'est le **parcours en profondeur**.

01 Le parcours en largeur (BFS)

> On utilise une **file** : on enfile le départ, puis tant que la file n'est pas vide, on défile un sommet et on enfile ses voisins **pas encore visités**.

```
from collections import deque

def largeur(g, depart):
    visites = [depart]
    file = deque([depart])
    while file:
        s = file.popleft()
        for v in g[s]:
            if v not in visites:
                visites.append(v)
                file.append(v)
    return visites
```

Le parcours en largeur trouve le chemin le plus court en nombre d'arêtes : les sommets sont atteints par distance croissante au départ.

Le parcours en profondeur (DFS)

On remplace la file par une **pile**, ou on écrit une fonction récursive :

© Astrolab Cours · astrolabcours.fr/nsi/terminale/parcours-de-graphes/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits

```python
def profondeur(g, s, visites=None):
    if visites is None:
        visites = []
    visites.append(s)
    for v in g[s]:
        if v not in visites:
            profondeur(g, v, visites)
    return visites
```

Il sert à savoir si un chemin existe, à détecter un cycle ou à explorer un labyrinthe.

À quoi ça sert

Question	Parcours
le plus court chemin en nombre d'étapes	largeur
un chemin existe-t-il ?	l'un ou l'autre
explorer à fond une branche, détecter un cycle	profondeur

Exemple guidé

Graphe : `g = {"A": ["B", "C"], "B": ["D"], "C": ["D", "E"], "D": [], "E": []}`. Distance de A à chaque sommet par un parcours en largeur.

```python
from collections import deque

def distances(g, depart):
    dist = {depart: 0}
    file = deque([depart])
    while file:
        s = file.popleft()
        for v in g[s]:
            if v not in dist:
                dist[v] = dist[s] + 1
                file.append(v)
    return dist

distances(g, "A")
# {'A': 0, 'B': 1, 'C': 1, 'D': 2, 'E': 2}
```

1. On visite d'abord les voisins de A (distance 1), puis leurs voisins (distance 2).
2. Le dictionnaire `dist` sert aussi de liste des **déjà vus** : on ne visite jamais deux fois un sommet.

Retour au problème

Le parcours en largeur donne le plus petit nombre d'intermédiaires entre deux personnes. Pour un labyrinthe, un parcours en largeur ou en profondeur dit s'il existe un chemin vers la sortie.

© Astrolab Cours · astrolabcours.fr/nsi/terminale/parcours-de-graphes/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits

Les pièges

- Oublier de marquer les sommets visités : boucle infinie dès qu'il y a un cycle.
- Le parcours en largeur donne le plus court chemin en **nombre d'arêtes**, pas en distance dans un graphe pondéré.
- L'ordre de visite dépend de l'ordre des voisins dans les listes.

© Astrolab Cours · astrolabcours.fr/nsi/terminale/parcours-de-graphes/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits