DSA · FICHE PRATIQUE

Cheatsheet : Graphes, BFS, DFS et Dijkstra

Choisir un parcours selon les arêtes et le résultat attendu.

Par Ali ZAINOUL · Cours gratuit

Définitions, contrats et coûts

Choisir un parcours selon les arêtes et le résultat attendu.

Graphes, BFS, DFS et Dijkstra

Notion	À retenir
Représentation	Listes d’adjacence : $\Theta(n + m)$ stockage ; matrice : $\Theta(n^2)$.
BFS / DFS	$\Theta(n + m)$ avec listes d’adjacence ; BFS minimise le nombre d’arêtes non pondérées.
Dijkstra	Poids non négatifs ; avec tas binaire, $O((n + m) \log n)$ dans le modèle usuel.

Vérifier sur un exemple

Distances en nombre d’arêtes

Le dictionnaire distance sert aussi d’ensemble des sommets déjà découverts.

Python

```python
# breadth_first_distances.py
from collections import deque
graph = {"A": ["B", "C"], "B": ["D"], "C": ["D"], "D": []}
distance = {"A": 0}
queue = deque(["A"])
while queue:
    vertex = queue.popleft()
    for neighbor in graph[vertex]:
        if neighbor not in distance:
            distance[neighbor] = distance[vertex] + 1
            queue.append(neighbor)
print(sorted(distance.items()))
# Expected output:
# [('A', 0), ('B', 1), ('C', 1), ('D', 2)]
```

Résultat expliqué. Un sommet est marqué lors de sa découverte, avant son ajout dans la file : les chemins concurrents ne le réinsèrent pas.

Erreur à éviter. BFS ne minimise pas la somme de poids arbitraires. Dijkstra n’accepte pas de poids négatifs.

▼ Un sommet inaccessible reçoit-il une distance finie dans ce programme ?

Non : il est absent de distance. L’interface doit distinguer absence de chemin et distance nulle.

Télécharger cette fiche en PDF · **Toutes les fiches DSA**

Version web : https://www.clearcode.fr/courses/programming/dsa/cheatsheet/graphs