TP N°4 - Parcours de Graphe

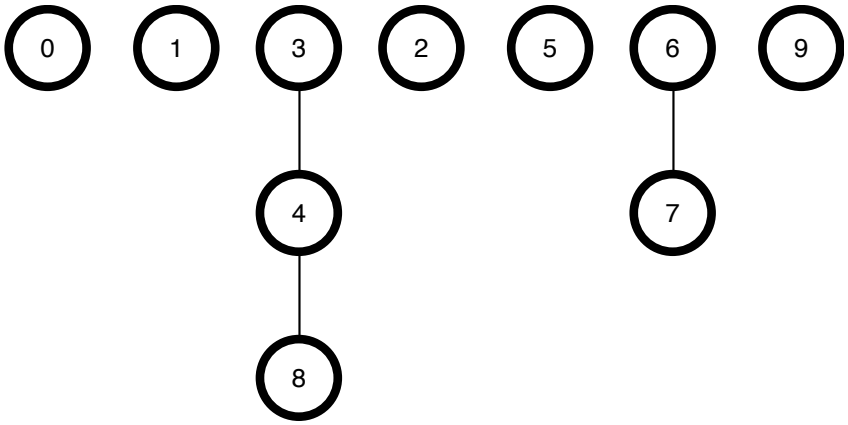
Préparé par : Benjamin DEBOTTÉ,, 1A App.Info

1 avril 2015

SOMMAIRE

I. CONCEPTION

En informatique les graphes sont utilisés comme structure de données, représentant une arborescence avec un système de branches. Certains arbres peuvent être qualifiés de connexes lorsque tout les éléments du graphes sont reliées et donc atteignables en un seul parcours et orienté si la navigation d'un noeud à l'autre peut être restreint par une direction.



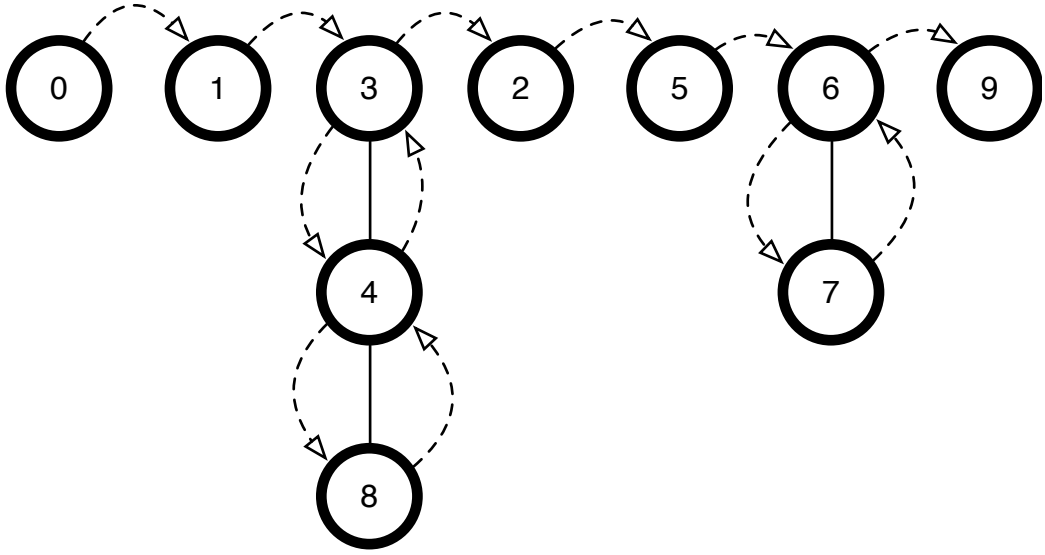
Ce TP consiste au développement des fonctions de parcours en largeur et en profondeur d'un graphe. Les deux méthodes de parcours consistent à boucler sur des visites de noeuds jusqu'à ce que l'ensemble des noeuds ait été visité. Seule la méthode de visite varie. Chaque méthode de parcours affiche le nombre de composantes connexes ainsi que l'affiliation de chaque noeud à une composante. Les composantes sont numérotés de 1 à n, avec n nombre de noeuds. Les composantes sont gérées à chaque appel de fonction de visite : un graphe connexe étant parcouru en un appel, un graphe non-connexe fera appel à la fonction pour chaque composante qu'il contient.

1. Visite en profondeur

La visite en profondeur est un algorithme récursif. Elle débute par une initialisation à l'état UNVISITED l'ensemble des cases d'un tableau représentant chacune un noeud du graphe.

Le noeud actuel est indiqué comme PENDING. On appelle récursivement la fonction avec tout les noeuds adjacents.

Quand la fonction arrive à la fin, le noeud actuel est marqué à VISITED. Récursivement, l'ensemble des noeuds seront marqués comme VISITED si le graphe est connexe. Sinon, la fonction de parcours rappellera la fonction de visite pour le premier noeud UNVISITED qu'elle croisera et en incrémentant le numéro de la composante connexe associée aux noeuds.



2. Visite en largeur

La visite en largeur est basée sur l'utilisation d'une file. Grâce à des enfilages successifs des noeuds adjacents, on obtient un parcours étage par étage. Si le graphe est connexe, tout les noeuds sont visités en une seule fois.

II. TRACE D'EXÉCUTION

Voici la trace associé au contenu du fichier graph_alea.txt :

```
10 3
4 8
4 3
7 6
```

Trace d'exécution :

```
# 0 1 2 3 4 5 6 7 8 9
0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0
3 0 0 0 0 1 0 0 0 0
4 0 0 0 1 0 0 0 0 1 0
5 0 0 0 0 0 0 0 0 0
6 0 0 0 0 0 0 0 1 0 0
7 0 0 0 0 0 0 1 0 0 0
8 0 0 0 0 1 0 0 0 0 0
9 0 0 0 0 0 0 0 0 0 0
Parcours en profondeur :
Noeud prefixe : 0
Noeud suffixe : 0
Noeud prefixe : 1
Noeud suffixe : 1
Noeud prefixe : 2
Noeud suffixe : 2
Noeud prefixe : 3
Noeud prefixe : 4
Noeud prefixe : 8
Noeud suffixe : 8
Noeud suffixe : 4
Noeud suffixe : 3
Noeud prefixe : 5
```

```
Noeud suffixe : 5
Noeud prefixe : 6
Noeud prefixe : 7
Noeud suffixe : 7
Noeud suffixe : 6
Noeud prefixe : 9
Noeud suffixe : 9
Nombre de noeuds visites : 10

Nombre de composants : 7
0 |1 |2 |3 |4 |5 |6 |7 |8 |9 |
1 |2 |3 |4 |4 |5 |6 |6 |4 |7 |
Parcours en largeur:
Noeud visité : 0
Noeud visité : 1
Noeud visité : 2
Noeud visité : 3
Noeud visité : 4
Noeud visité : 8
Noeud visité : 5
Noeud visité : 6
Noeud visité : 7
Noeud visité : 9
Nombre de noeuds visites : 10

Nombre de composants : 7
0 |1 |2 |3 |4 |5 |6 |7 |8 |9 |
1 |2 |3 |4 |4 |5 |6 |6 |4 |7 |
```