

Algorithmique (AL5)

TD n° 2 : parcours en largeur de graphes non orientés

Les exercices notés avec une étoile (*) présentent un niveau de difficulté plus élevé.

Exercice 1 : parcours en largeur

Appliquer aux deux graphes ci-dessous l'algorithme de parcours en largeur à partir du sommet 1. Pour chaque graphe, donner l'arbre résultant de ce parcours.

Parcours en largeur (BFS ¹)

Entrées : graphe $G = (V, E)$ et sommet $s \in V$

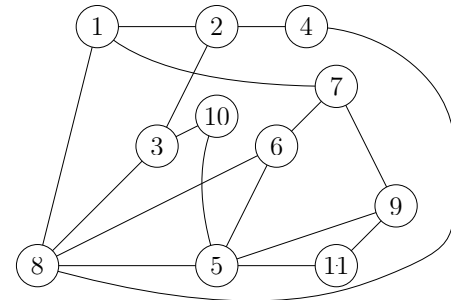
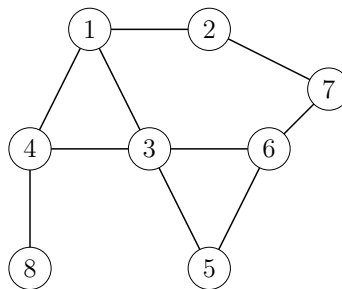
début

```

    créer file( $Q$ ) ;
    marquer( $s$ ) ;
    enfiler( $Q, s$ ) ;
    tant que  $Q \neq \emptyset$  faire
         $u \leftarrow$  défiler( $Q$ ) ;
        pour tous les  $uv \in E$  faire
            si  $v$  non marqué alors
                marquer( $v$ ) ;
                enfiler( $Q, v$ )

```

1. Breadth-first search



Algorithme de parcours en largeur

Graphe G_1

Graphe G_2

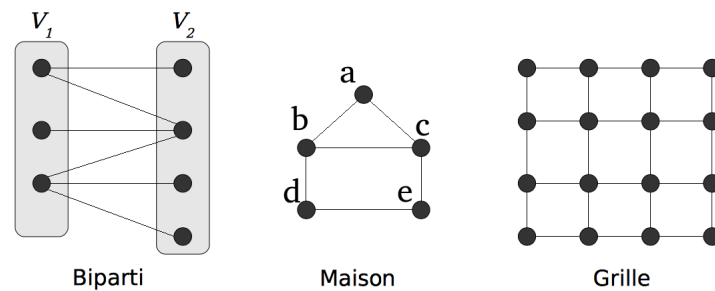
Exercice 2 : parcours et matrice d'adjacence

Quelle est la complexité de l'algorithme de parcours en largeur d'un graphe non orienté si celui-ci est représenté par une matrice d'adjacence ?

Exercice 3 : plus courts chemins

Dans cet exercice $G := (V, E)$ est un graphe quelconque, et $s \in V$, un de ses sommets. On a $(|V|, |E|) = (n, m)$.

1. Proposer un algorithme de complexité en $O(n + m)$ (nombre de sommets plus nombre d'arêtes) qui calcule la distance s et les autres sommets de G .
2. En déduire un algorithme de complexité en $O(n * m)$ qui calcule un des chemin le plus court entre s et chacun des autres sommets de G .
3. (bonus) En proposant une manière astucieuse d'encoder la sortie, en déduire un algorithme de complexité en $O(n + m)$ qui calcule un chemin le plus court entre s et chacun des autres sommets de G .

FIGURE 1 – Schéma général d'un graphe biparti, la maison, la grille $G_{4,4}$ **Exercice 4 : graphes bipartis**

Un graphe $G = (V, E)$ est un graphe biparti si l'ensemble des sommets V peut être partitionné en deux sous-ensembles V_1 et V_2 (i.e. $V_1 \cap V_2 = \emptyset$ et $V_1 \cup V_2 = V$), de sorte que les arêtes de E ont exactement une extrémité dans V_1 et une extrémité dans V_2 .

1. Déterminer si le graphe maison est un graphe biparti.
2. Même question pour la grille $G_{4,4}$ (on pourra nommer les sommets (i, j) selon leur coordonnées).
3. Montrer qu'un graphe est biparti si et seulement si il n'a pas de cycle de longueur impaire. La longueur d'un cycle est le nombre d'arêtes qui le composent. (Astuce : on peut se servir d'une 2-coloration).

Remarque : colorier un graphe revient à mettre des couleurs sur chaque sommet du graphe. La seule contrainte imposée est que deux sommets adjacents ne soient pas de la même couleur.

4. Proposer un algorithme qui prend en entrée un graphe, qui teste si le graphe est biparti, et qui renvoie un cycle impair sinon. On pourra se servir du parcours en largeur (BFS) vu en cours.
5. Évaluer la complexité de votre algorithme.

Exercice 5 : cycle (*)

Proposer un algorithme renvoyant, s'il existe, un cycle de longueur minimale passant par un sommet s donné dans un graphe G non orienté. Prouver sa correction. On pourra dans un premier temps supposer que le graphe est bipart