TP2-Théorie des graphes et Algorithmie(Python)

Master 1 OIVM-UPEC

Remarques !!

Il est demandé aux étudiants de rendre un compte rendu du TP à la fin de chaque semaine (au plus tard samedi à 23H59)

Parcours en profondeur

Exercice1

Écrire un programme qui assure le parcours en profondeur d'un graphe .

Expliquer comment le marquage (blanc/gris/noir), de chaque sommet, est stocké durant le parcours (Pensez à commenter le code !)

Exercice 2

Améliorer le programme précédent pour mémoriser certaines informations utiles à propos de chaque sommet :

- Son prédécesseur dans le parcours
- Les dates auxquelles il passe de blanc à gris et de gris à noir (on maintiendra un compteur de date qui augmentera de 1 à chaque fois qu'un sommet change de couleur).

Parcours en largeur

Exercice 3

Écrire un programme qui assure le parcours en largeur d'un graphe. Pour la file, on pourra se contenter d'une liste Python, en utilisant F.pop(0) pour supprimer l'élément de tête.

Exercice 4

Améliorer également ce programme pour mémoriser à propos de chaque sommet :

- Son prédécesseur dans le parcours ;
- Sa distance à l'origine du parcours.

Autres Algorithmes

Exercice 5

Le problème du tri topologique est un problème d'ordonnancement des tâches : il y a une liste de choses à faire, certaines ont besoin d'être faites avant d'autres. On représente ces contraintes par un graphe orienté dont les sommets sont les tâches et les arêtes sont les couples (u, v) tels que la tâche u doit être effectuée avant la tâche v. On cherche alors un ordre sur les sommets tel que si (u, v) est une arête, alors u apparait avant v dans l'ordre. Le problème n'a pas de solution s'il y a un cycle dans le graphe, on suppose donc le graphe acyclique.

Une façon de réaliser un tri topologique d'un graphe G est de faire un parcours en profondeur, et de placer les sommets dans la liste à la fin de leur visite (contrairement au parcours en profondeur qui les place dans la liste au début de leur visite)

Écrire un programme qui permet d'effectuer un tri topologique sur un graphe, basé sur les dates de passage en noir dans un parcours en profondeur.

Exercice 6

Écrire un programme qui permet d'énumérer des composantes fortement connexes, dont on rappelle le principe général :

- Calculer un tri topologique du graphe G
- Calculer le graphe transposé G^T ;
- Effectuer un parcours en profondeur dans G^T , en choisissant les sources s dans l'ordre du tri topologique de G : tous les sommets atteints à partir de s forment la composante fortement connexe contenant s.

Indication : Il est conseillé d'utiliser un dictionnaire qui, à chaque sommet, associe la liste des sommets de sa composante fortement connexe.

Exercice 7

Écrire un programme qui construit le graphe quotient de G par ses composantes fortement connexes.

Une composante fortement connexe d'un graphe G est un sous-graphe maximal connexe de G.