CentraleSupélec

Algorithmique et complexité

TD 1/7 – Parcours de graphes

Ce TD est l'occasion de mettre en œuvre les premiers algorithmes vus en cours, les parcours de graphes.

Exercice 1 : Parcours en largeur et graphes bipartis

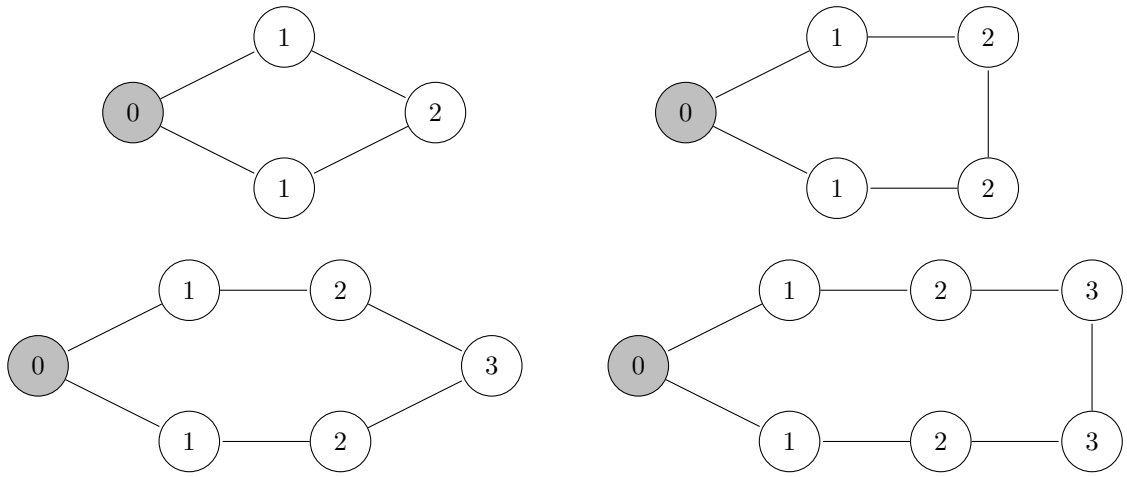
Première partie – Profondeur des sommets et cycle de longueur impaire

Question 1

En vous appuyant sur les algorithmes de parcours de graphe vus en cours, proposez un algorithme qui calcule la distance en nombre d'arêtes entre un sommet du graphe (désigné comme racine) et tous les autres sommets (on parle de *profondeur*).

Question 2

Considérons les cycles suivants de longueur paires et impaires pour lesquels nous avons calculé les profondeurs :



Que remarquez vous (sur les profondeurs) dans les cas des graphes avec un cycle **impair** (en nombre d'arêtes) ? Est-ce une propriété caractéristique ? Énoncez le cas général et donnez une preuve !

Question 3

Proposez un algorithme qui détermine si un graphe contient un cycle impair.

Deuxième partie – Graphe biparti

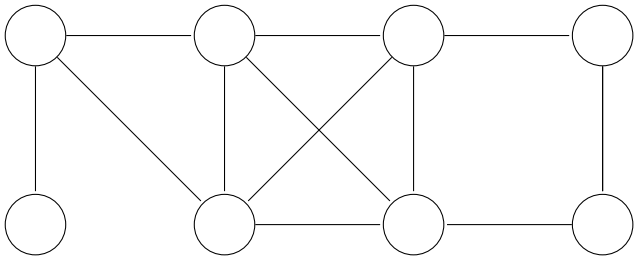
Un graphe $G = (V, E)$ est dit biparti s'il existe une partition de son ensemble de sommets en deux sous-ensembles $V_1 \subseteq V$ et $V_2 \subseteq V$ telle que chaque arête de E a une extrémité dans V_1 et l'autre dans V_2.

Question 4

Dans un graphe biparti, peut-t-il exister un cycle avec un nombre impair d'arêtes ? est ce une propriété caractéristique ? Justifiez votre réponse.

Question 5

Déduisez un algorithme qui permet de déterminer si un graphe est biparti. Testez votre algorithme sur le graphe suivant. Est-il biparti ? Justifiez votre réponse



Exercice 2 : Parcours en profondeur et graphes 2-coloriables

Le coloriage de graphe consiste à attribuer une couleur à chacun de ses sommets de manière à ce que deux sommets reliés par une arête soient de couleurs différentes.

Un graphe 2-coloriable est un graphe pouvant être colorié avec seulement 2 couleurs.

Question 1

Quel est le lien avec l'exercice précédent ? Justifiez votre réponse.

Question 2

Nous souhaitons écrire un algorithme, inspiré du parcours en **profondeur** vu en cours, qui prend en entrée un graphe `G` et qui renvoie un couple `(result,color)` où `result` est `True` si le graphe est coloriable, `False` sinon et `color` est un dictionnaire associant une couleur 0 ou 1 à chaque sommet.

Cet algorithme devra **s'arrêter au plus tôt** lorsque le graphe n'est pas 2-coloriable.

Proposez une version **itérative** et une version **récursive**.

Question 3

Est ce que votre algorithme est correct, c'est-à-dire :

1. Quand il retourne un coloriage, ce 2-coloriage est-il correct ?
2. Quand il retourne `False`, est-on certain que le graphe n'est pas 2-coloriable ?

Question 4

Testez votre algorithme sur le graphe ci-dessus. Est-il 2-coloriable ?

Exercice 3 : Une propriété des DAGs

Prouvez le théoreme suivant :

Dans un graphe orienté sans circuit (c'est à dire un DAG – *Directed Acyclic Graph*) il existe au moins un sommet sans arcs entrants.