TD sur les graphes

Le code sera rédigé exclusivement en langage Python.
On pourra se connecter à l'activité Capytale numéro 92b5-812071.

Note : toute fonction demandée par le sujet peut être utilisée dans la suite.

1 Problème

Sujet adapté d'une épreuve de L. Gonnord pour Univ. Lyon 1.

2 Définitions

Un graphe $G = (V, E)$ est dit **biparti** si on peut partitionner son ensemble de sommets V en deux sous-ensembles A et B distincts, non vides, de sorte que toute arête ait une extrémité dans A et une extrémité dans B.
Si les ensembles A et B ont le même cardinal, on dit qu'il s'agit d'un graphe biparti équilibré.
Dans tout le problème, on ne considère que des graphes bipartis équilibrés.
On note n le cardinal commun aux ensembles A et B ; la taille du graphe est donc égale à $2n$.

On suppose que l'on a toujours $n \geq 1$.
Les sommets de A sont numérotés de 0 à $n-1$ et nommés $0_A, 1_A, 2_A, \ldots, (n-1)_A$; les sommets de B sont numérotés de 0 à $n-1$ et nommés $0_B, 1_B, 2_B, \ldots, (n-1)_B$.
Une arête de G est toujours écrite en mettant d'abord l'extrémité qui est dans A puis celle qui est dans B.

On représente les graphes bipartis équilibrés par des schémas comme on peut le voir dans la figure 1 , en représentant les sommets de A à gauche et les sommets de B à droite.

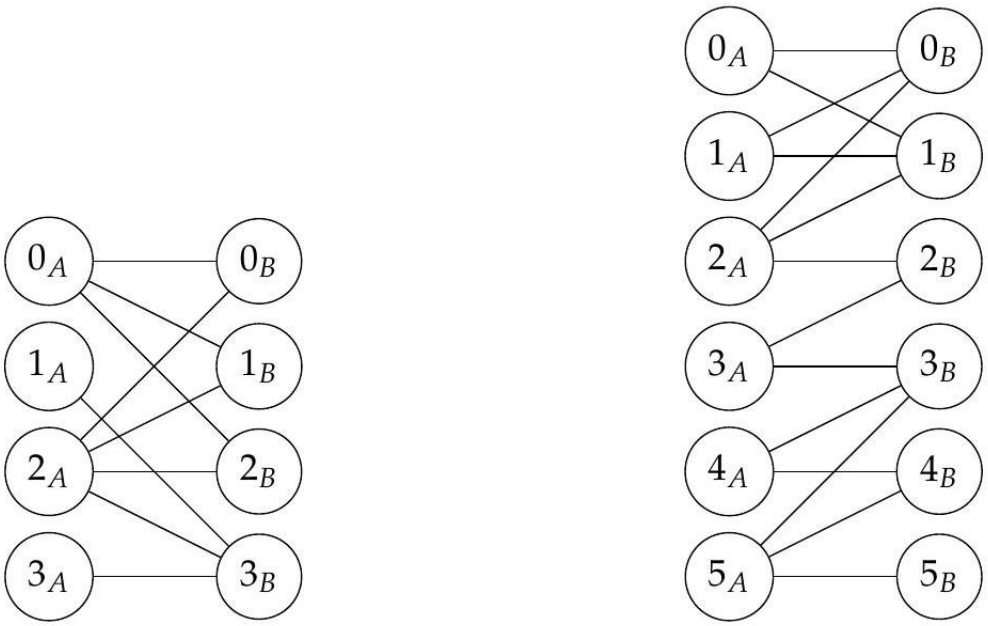


Figure1 : Le graphe G_0 est représenté à gauche, le graphe G_1 à droite.

On dit que deux arêtes d'un graphe G sont **incidentes** si elles ont une extrémité en commun.
On appelle couplage dans G un ensemble d'arêtes de G deux à deux non incidentes.

2.1 Première partie : généralités

Pour le moment, la représentation du graphe se fait à l'aide d'une matrice d'adjacence de taille $2n \times 2n$ dans laquelle les sommets de l'ensemble A sont placés avant ceux de l'ensemble B.

1. Déterminer la matrice d'adjacence du graphe G_0. Pour cette représentation, on notera la présence d'une arête entre deux sommets par la valeur 1.

   Dans la suite, on considérera que la matrice d'adjacence contient les valeurs True ou False.
2. Donner un couplage de cardinal 3 dans G_0.
3. Indiquer s'il existe dans G_0 un couplage de cardinal 4. Justifier la réponse.

   Un couplage est représenté par une liste d'entiers indexée de 0 à $n-1$. Soit i vérifiant $0 \leq i \leq n-1$; :

- si le sommet i_A est couplé avec le sommet j_B, la case d'indice i contient la valeur j ;
- si le sommet i_A n'est pas couplé, la case d'indice i contient une valeur égale à -1.

Par exemple, si $n = 4$, la liste $[2; -1; 3; -1]$ représente la couplage $\{0_A - 2_B; 2_A - 3B\}$.

4. Quel tableau représente le couplage de cardinal 3 donné en réponse à la question 2?

Représentation de la matrice d'adjacence
Dans la suite du sujet, la représentation du graphe sera donnée par une matrice d'adjacence de taille $n \times n$ avec en ligne les indices de 0 à $n-1$ des sommets de l'ensemble A et en colonne les indices de 0 à $n-1$ des sommets de l'ensemble B.

5. Écrire en Python une fonction ***verifie(G, C, n)*** qui, étant donné le graphe G codé sous la forme d'une matrice d'adjacence G de taille $n \times n$, renvoie True si le tableau C représente un couplage C dans G, False sinon.

   Justifier rapidement la complexité de ***verifie***.

6. Écrire en Python une fonction ***cardinal*** (C, n) qui, étant donné un couplage C entre deux ensembles de taille n, renvoie le cardinal de ce couplage.
   Justifier rapidement la complexité de cardinal.

2.2 Deuxième partie : un algorithme pour déterminer un couplage maximal

On dit qu'un couplage C dans un graphe G est **maximal** si toute arête de G n'appartenant pas à C est incidente à au moins une arête de C.
Un couplage maximal de G n'est pas forcément de cardinal maximum parmi les couplages de G.

7. Sur le graphe G_0, donner un couplage maximal de cardinal 2.

On cherche à concevoir un algorithme qui détermine un couplage maximal dans un graphe biparti équilibré G.

Algorithme Approché :

— on commence avec un couplage vide C ;

— tant que G possède au moins une arête :
— on choisit une arête a de G dont la somme des degrés des extrémités soit minimum ;
— on ajoute l'arête a au couplage C ;
— on retire de G l'arête a et toutes les arêtes incidentes à a.

On admettra que le résultat est, par construction, un couplage maximal.

8. Appliquer l'algorithme au graphe G_0.

Étude du graphe G_1
On considère par la suite le graphe biparti équilibré G_1 dont le cardinal des ensembles A et B est 6 . Ce graphe est représenté à droite sur la figure 1.

9. On applique l'algorithme au graphe G_1. Déterminer la première arête a_1 choisie ; tracer le graphe obtenu après suppression de a_1 et des arêtes incidentes à a_1.
Montrer que le couplage obtenu par cet algorithme est de cardinal au plus 5 et indiquer s'il est de cardinal maximum parmi les couplages de G_1.

Représentation d'une arête : on représentera en Python une arête par un couple d'entiers représentant les numéros de ces deux extrémités.

10. Écrire une fonction ***arete_min*** (G, n) qui détermine une arête dont la somme des degrés des extrémités est minimale dans un graphe biparti équilibré G représenté par une matrice de taille $n \times n$. La fonction retourne False, None si le graphe ne possède pas d'arêtes. Sinon la fonction retourne True et la représentation d'une arête. Indiquer la complexité de la fonction ***aretemin(G)***.
11. Écrire une fonction ***supprimer_arete*** (G, n***, a)*** qui supprime d'un graphe G représenté par une matrice d'adjacence de taille $n \times n$ une arête a représentée par un couple d'entiers et toutes les arêtes incidentes à a. Indiquer la complexité de la fonction supprimer_arete.
12. Écrire une fonction ***copier*** (G, n) qui renvoie une copie profonde d'un graphe G représenté par une matrice d'adjacence de taille $n \times n$.
13. Écrire une fonction ***algo_approche(G)*** qui, à partir d'une matrice G codant un graphe biparti équilibré G, applique l'algorithme à partir d'une copie de G et renvoie un tableau codant le couplage obtenu. Indiquer la complexité de la fonction ***algo_approche***.

3 Troisième partie : recherche exhaustive d'un couplage de cardinal maximum

De la même manière que dans la deuxième partie, on représentera en Python une arête par un couple d'entiers représentant les numéros de ces deux extrémités.

Rappel : Un graphe biparti équilibré G est représenté par une **matrice d'adjacence** de taille $n \times n$ avec en ligne les éléments de 0 à $n-1$ de l'ensemble A et en colonne les éléments de 0 à $n-1$ de l'ensemble B.

14. Soit G un graphe biparti équilibré représenté par une matrice d'adjacence de taille $n \times n$.
Écrire une fonction ***une_arete*** (G, n) qui recherche une arête quelconque de G.
Si G possède au moins une arête, la fonction renvoie True et la première arête rencontrée a sous la forme d'un couple. Sinon la fonction renvoie (False, None). Donner la complexité de la fonction ***une_arete***.

On cherche à établir un algorithme récursif, nommé meilleur_couplage qui permette de déterminer un couplage de cardinal maximum dans un graphe biparti équilibré :

Si le graphe courant ne contient aucune arête, le cardinal maximum d'un couplage est 0 et aucun sommet n'est couplé. Dans le cas contraire, l'algorithme considère une arête quelconque a du graphe courant et recherche successivement :

- un couplage de cardinal maximum parmi les couplages du graphe courant ne contenant pas a
- un couplage de cardinal maximum parmi les couplages du graphe courant contenant a

L'algorithme déduit alors un couplage de cardinal maximum.

15. Écrire une fonction ***récursive meilleur_couplage*** (G, n) qui renvoie un tableau codant un couplage de cardinal maximum dans G en utilisant l'algorithme ci-dessus.
Donner la complexité de ***meilleur_couplage***.