MINISTÈRE
DE L'ÉDUCATION
NATIONALE,
DE LA JEUNESSE
ET DES SPORTS

Liberté
Égalité
Fraternité

VOIE GÉNÉRALE

2DE | 1RE | TLE

Numérique et sciences informatiques

ENSEIGNEMENT
SPÉCIALITÉ

REPRÉSENTATION DES GRAPHES

Cette ressource traite de la représentation des graphes dans un ordinateur. Il en existe de nombreuses, mais cette étude porte sur les deux plus courantes : celle par *matrice d'adjacence* et celle par *listes d'adjacences*. Le choix de la représentation dépend de la nature du graphe, des algorithmes que l'on veut utiliser et du langage de programmation utilisé pour l'implémentation.

Dans cette étude, on considère un graphe $G = (S, A)$, orienté ou non, vérifiant les caractéristiques suivantes :

- S contient n sommets numérotés de 1 à n ;
- chaque sommet a au plus une boucle ;
- il existe au plus une arête/un arc entre deux sommets.

Matrice d'adjacence

Définitions et exemples

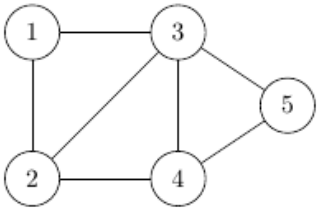
Définition 1 - Cas des graphes non valués

On appelle *matrice d'adjacence* du graphe G un tableau M à n lignes et n colonnes, où le coefficient $M_{i,j}$ situé à la i ème ligne et j ème colonne est défini par :

$$M_{i,j} = \begin{cases} 1 & \text{si } (i,j) \in A \\ 0 & \text{si } (i,j) \notin A \end{cases}$$

Exemple 1

La figure 1b donne la matrice d'adjacence du graphe de la figure 1a :



(a) Graphe

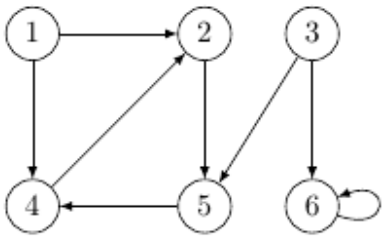
	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	1	0
3	1	1	0	1	1
4	0	1	1	0	1
5	0	0	1	1	0

(b) Matrice d'adjacence

FIGURE 1 – Un graphe non orienté et sa représentation en matrice d'adjacence

Exercice 1

Donner la matrice d'adjacence du graphe de la figure 2a ci-dessous :



(a) Graphe (b) Matrice d'adjacence

FIGURE 2 – Un graphe orienté et sa représentation en matrice d'adjacence

Remarques

- Comme la relation d'adjacence entre deux sommets est symétrique dans un graphe non orienté, la matrice d'adjacence correspondante est symétrique par rapport à sa diagonale principale.
- La définition 1 peut être étendue facilement aux multigraphes en disant que la valeur $M_{i,j}$ correspond aux nombres d'arêtes/d'arcs partant du sommet i et arrivant au sommet j.

Dans le cas des graphes valués, la définition 1 peut être aisément adaptée en remplaçant le nombre d'arêtes/d'arcs entre deux sommets i et j par le coût $C(i,j)$ de l'arête/arc (i,j). On a alors la définition 2.

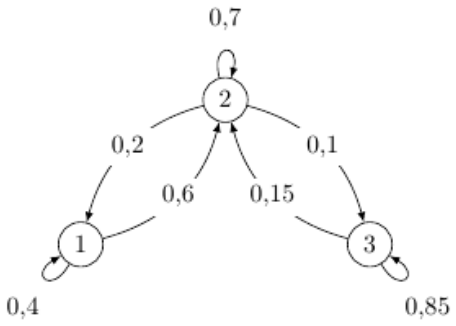
Définition 2 - Cas des graphes valués

On appelle matrice d'adjacence du graphe G un tableau M à n lignes et n colonnes, où le coefficient $M_{i,j}$ situé à la i ème ligne et j ème colonne est défini par :

$$M_{i,j} = \begin{cases} C(i,j) & \text{si } (i,j) \in A \\ +\infty & \text{si } (i,j) \notin A \end{cases}$$

Exercice 2

Donner la matrice d'adjacence du graphe de la figure 3a ci-dessous :



(a) Graphe (b) Matrice d'adjacence

FIGURE 3 – Un graphe valué et sa représentation en matrice d'adjacence

Remarques

- Dans la définition 2, on utilise également couramment à la place de la valeur $+\infty$ la valeur 0 ou la constante NIL.
- Dans le cas particulier d'un graphe probabiliste, la matrice d'adjacence est plus précisément appelée *matrice de transition* ou *matrice stochastique*. La matrice de l'exercice 2 est une matrice stochastique.

Avantages et inconvénients

La représentation d'un graphe à l'aide d'une matrice d'adjacence a de nombreux avantages :

- elle est facile à construire ;
- on a un accès rapide à une arête/un arc donné(e) et aux voisins d'un sommet donné (temps constant).

De plus, dans le cas des graphes non valués, on a le résultat théorème 1.

Théorème 1

Soient M la matrice d'adjacence d'un graphe non valué et $\mathrm{p} \geq 1$ un entier. Alors, le coefficient situé à la i ème ligne et j ème colonne de la matrice M^{p} est égal au nombre de chemins de longueur p partant du sommet i et arrivant au sommet j.

Exemple 2

Reprenons la matrice M de l'exemple 1. On a alors :

$$M^2 = \begin{pmatrix} 2 & 1 & 1 & 2 & 1 \\ 1 & 3 & 2 & 1 & 2 \\ 1 & 2 & 4 & 2 & 1 \\ 2 & 1 & 2 & 3 & 1 \\ 1 & 2 & 1 & 1 & 2 \end{pmatrix} \text{ et } M^3 = \begin{pmatrix} 2 & 5 & 6 & 3 & 3 \\ 5 & 4 & 7 & 7 & 3 \\ 6 & 7 & 6 & 7 & 6 \\ 3 & 7 & 7 & 4 & 5 \\ 3 & 3 & 6 & 5 & 2 \end{pmatrix}$$

et on peut en déduire, par exemple, les informations suivantes :

- il y a deux chemins de longueur 2 entre les sommets 3 et 4 et trois chemins fermés de longueur 2 et d'extrémités le sommet 4 ;
- il y a cinq chemins de longueur 3 entre les sommets 1 et 2 et six chemins fermés de longueur 3 et d'extrémités le sommet 3.

Exercice 3

Notons M la matrice d'adjacence du graphe de l'exercice 1. On a alors :

$$M^2 = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \text{ et } M^4 = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

- Existe-t-il des chemins de longueur 2 entre les sommets 1 et 4 ? 4 et 5 ? Si oui, combien ?
- Existe-t-il des chemins de longueur 4 entre les sommets 2 et 5 ? 6 et 6 ? Si oui, combien ?

Remarque

Dans le cas des graphes valués, il n'y a pas d'énoncés analogues à celui du théorème 1, sauf si le graphe est un graphe probabiliste.

Le principal inconvénient de la représentation d'un graphe par matrice d'adjacence est, bien sûr, la taille mémoire, puisqu'il faut gérer n^2 cases mémoires pour représenter un graphe à n sommets. De plus, le calcul des puissances de la matrice d'adjacence a un coût mémoire non négligeable. Par conséquent, ce mode de représentation est en général utilisé uniquement pour les graphes de petite taille et les graphes denses, c'est-à-dire lorsque $\text{card}(A)$ est proche de $\left(\text{card}(S)\right)^2$, ou quand on veut savoir rapidement s'il existe une arête/un arc entre deux sommets donnés.

Implémentation

En `Python`, la matrice d'adjacence d'un graphe est en général implémentée à l'aide de listes de listes ou de tableaux `numpy` (importer le module `numpy` à l'aide de l'instruction `import numpy`).

Exemple 3

Reprenons la matrice M de l'exemple 1. On a les deux implémentations suivantes :

```
M = [[0,1,1,0,0],
     [1,0,1,1,0],
     [1,1,0,1,1],
     [0,1,1,0,1],
     [0,0,1,1,0]]
```

(a) Implémentation à l'aide d'une liste de listes

```
M = numpy.array([[0,1,1,0,0],
                 [1,0,1,1,0],
                 [1,1,0,1,1],
                 [0,1,1,0,1],
                 [0,0,1,1,0]])
```

(b) Implémentation à l'aide d'un tableau `numpy`

FIGURE 4 – Deux implémentations de la matrice d'adjacence M

Quelques observations :

- dans l'implémentation de la figure 4a, l'instruction `len(M)` donne le nombre de sommets ;
- dans l'implémentation de la figure 4b, la taille du tableau est donnée par `numpy.shape(M)` qui renvoie le tuple `(5,5)` ;
- pour accéder, par exemple, à l'élément situé à la ligne i et colonne j, on saisit l'instruction `M[i][j]` dans les deux cas.

Remarques

- L'un des avantages d'utiliser une implémentation à l'aide de tableaux `numpy` est que l'on peut effectuer facilement le produit matriciel. Par exemple, pour calculer M^2, on utilise l'instruction `numpy.dot(M,M)` qui renvoie un tableau `numpy`.
- Dans les deux représentations ci-dessus, attention au fait que les indices `i` et `j` commencent à `0`.

Exercice 4

On considère la matrice d'adjacence M du graphe de l'exercice 2.

- Donner l'implémentation de M à l'aide d'une liste de listes et à l'aide d'un tableau `numpy`.
- Quelle instruction faut-il saisir pour connaître le nombre de sommets du graphe ?
- Quelle instruction faut-il saisir pour savoir si le sommet 2 est adjacent au sommet 4 ?

Listes d'adjacences

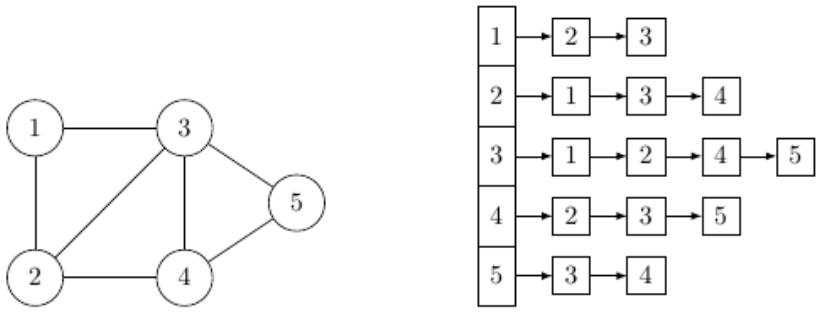
Définition et exemples

Définition 3 - Cas des graphes non valués

On appelle *listes d'adjacences* du graphe G un tableau Adj de n listes, la i ème liste contenant la liste des sommets adjacents au sommet i (l'ordre étant purement arbitraire). Autrement dit, le tableau Adj code les arêtes/arcs du graphe G.

Exemple 4

La figure 6b donne une représentation par listes d'adjacences du graphe de la figure 6a :

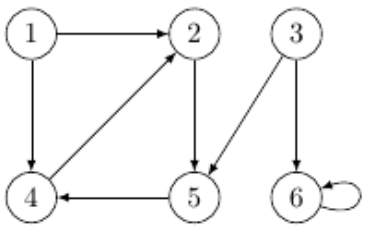


(a) Graphe (b) Listes d'adjacences

FIGURE 6 – Un graphe non orienté et sa représentation en listes d'adjacences

Exercice 5

Donner une représentation par listes d'adjacences du graphe de la figure 7a ci-dessous :



(a) Graphe (b) Listes d'adjacences

FIGURE 7 – Un graphe orienté et sa représentation en listes d'adjacences

Remarque

Dans le cas des graphes valués, le coût $C(i,j)$ de l'arête/arc est stocké sous forme d'attribut avec le sommet j dans la liste d'adjacence de i si $(i,j) \in A$.

Avantages et inconvénients

La représentation des graphes à l'aide de listes d'adjacences a de nombreux avantages :

- elle est compacte : on code uniquement les arêtes/arcs présents dans le graphe ;
- elle est robuste : on peut facilement modifier/ajouter des attributs pour prendre en charge les différents types de graphe.

Toutefois, elle admet également plusieurs inconvénients potentiels. Par exemple, pour déterminer si une arête/un arc donné (i, j) est présent dans le graphe, il n'existe pas de moyen plus rapide que de rechercher j dans la liste d'adjacence $Adj[i]$ de i. Par conséquent, ce mode de représentation est en général utilisé pour les graphes peu denses, c'est-à-dire lorsque $\text{card}(A)$ est très inférieur à $\left(\text{card}(S)\right)^2$.

Implémentation

En `Python`, les listes d'adjacences d'un graphe peuvent être implémentées à l'aide d'un dictionnaire, les clés étant les sommets et les valeurs des listes, éventuellement vides, contenant les sommets adjacents associés à chacune des clés.

Exemple 5

Reprenons les listes d'adjacences du graphe de l'exemple 4. On a l'implémentation suivante :

```
Adj = {1:[2,3], 2:[1,3,4], 3:[1,2,4,5], 4:[2,3,5], 5:[3,4]}
```

Exercice 6

Donner l'implémentation des listes d'adjacences des graphes des exercices 2 et 5 à l'aide d'un dictionnaire.