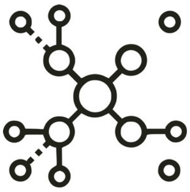
Les graphes



De manière générale, un graphe permet de représenter la structure, les connexions d'un ensemble complexe en exprimant les relations entre ses éléments : réseau de communication, réseaux routiers, interaction de diverses espèces animales, circuits électriques, …
L'histoire de la théorie des graphes débute avec les travaux d'Euler au XVIIIe siècle. La théorie des graphes s'est alors développée dans diverses disciplines telles que la chimie, la biologie, les sciences sociales. Les graphes constituent donc une méthode de pensée qui permet de modéliser une grande variété de problèmes en se ramenant à une modélisation commune. Les derniers travaux en théorie des graphes sont souvent effectués par des informaticiens, du fait de l'importance qu'y revêt l'aspect algorithmique.

1. Les graphes

Un graphe est une structure de données constituée d'objets appelés sommets et de relations entre ces sommets. Un graphe est généralement dessiné par des cercles représentant les sommets et par des traits reliant ces cercles pour représenter les relations liant les sommets.
Il existe deux types de graphes : les graphes orientés et les graphes non orientés

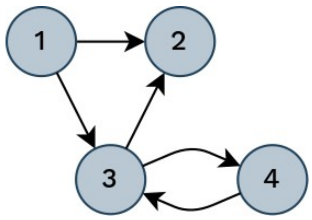


Figure 1: Graphe orienté

Graphes orientés

Une graphe orienté possède des relations orientées entre les sommets. Ces relations sont alors appelées des arcs et sont dessinées avec des flèches afin d'indiquer leur sens.

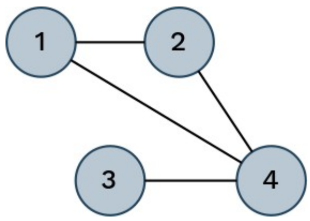


Figure 2: Graphe non orienté

Graphes non orientés

Dans un graphe non orienté le sens des relations n'est pas significatif. Les relations sont alors appelées des arêtes.

✗ Deux arêtes d'un graphe sont dites **adjacentes** si elles possèdent au moins un sommet en commun.

✗ Deux sommets d'un graphe non orienté sont dits **adjacents** s'il existe une arête les joignant. Des sommets adjacents sont aussi appelés **voisins**.

✗ Une **chaîne** ou un **chemin** est une suite consécutive d'arêtes sur un graphe non orienté.

✗ Lorsqu'un graphe non orienté est en un seul morceaux, c'est-à-dire lorsqu'il existe pour tous les sommets du graphe au moins un chemin les reliant, alors le graphe est dit **connexe**. Par exemple le graphe figure 2 est connexe.

2.Représentation des graphes

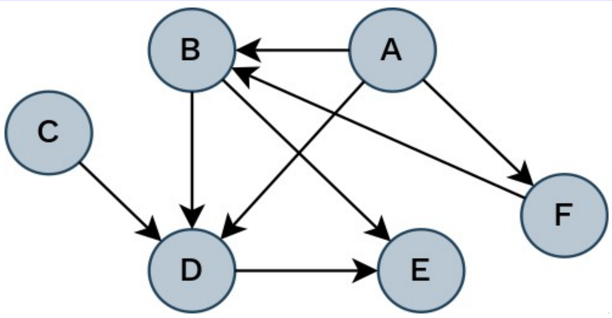
Deux modes de représentation distinctes peuvent être implémentées pour stocker un graphe :

- ➔ les matrices d'adjacence
- ➔ Les listes d'adjacences appelées liste des voisins pour les graphes non orientés ou liste des successeurs et des prédécesseurs pour les graphes orientés.

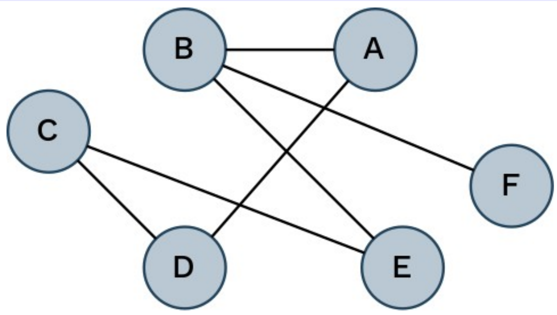
Matrice d'adjacence

Une matrice d'adjacence d'un graphe non orienté est un tableau de booléens dont la taille dépendra du nombre de sommets dans le graphe. Pour un nombre N de sommets, la matrice d'adjacence aura une taille N×N c'est à dire un tableau de N lignes et N colonnes.
La valeur booléenne contenue dans le tableau à la ligne **i**, colonne **j** indique la présence d'un arc entre les sommets **i** et **j** du graphe.
Les exemples suivants apportent une illustration de cette table d'adjacence dans le cas d'un graphe orienté et dans le cas d'un graphe non orienté :

Graphe A



Graphe B



Matrices d'adjacences

$$A=\begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

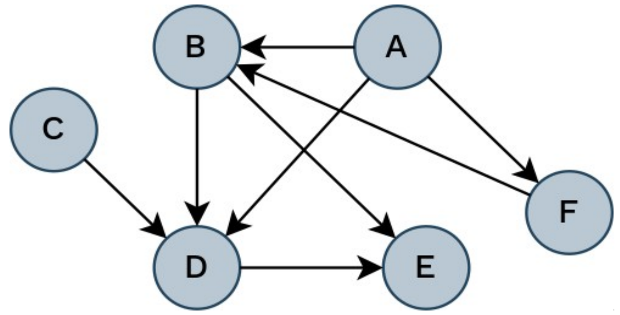
$$B=\begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Efficacité :
La matrice d'adjacence est simple à mettre en œuvre mais elle a néanmoins des défauts. D'une part elle occupe un espace proportionnel à NxN. Ainsi un graphe de mille sommets nécessite un graphe d'un million de booléens quelque soit le nombre de relations dans le graphe. D'autre part, parcourir tous les voisins d'un sommet nécessite de parcourir toute une ligne de la matrice alors même qu'il peut y avoir très peu de voisins.

Liste d'adjacence

Les défauts de la matrice d'adjacence peuvent être éliminés par la définition d'une liste d'adjacence. Dans ce cas, une liste de voisins est associée à chaque sommet. La représentation des graphes A et B précédents sera donc :

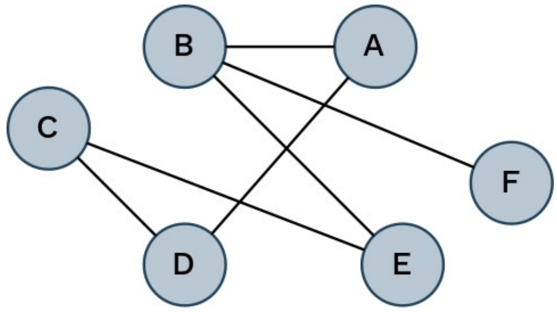
Graphe A



Listes des successeurs / prédécesseurs

Sommet	*Liste des successeurs*	*Liste des prédécesseurs*
A	(B, D, F)	Ø
B	(D, E)	(A, F)
C	(D)	Ø
D	(E)	(A, B, C)
E	Ø	(B,D)
F	(B)	(A)

Graphe B



Liste des voisins

Sommet	*Liste des voisins*
A	(B, D)
B	(A, F, E)
C	(E, D)
D	(C, A)
E	(B, C)
F	(B)

<u>Efficacité :</u>

Cette liste est souvent implémentée à l’aide d’un dictionnaire, où les clés portent le nom des sommets. On parle alors de dictionnaire d’adjacence.

Cette structure a pour avantage d’avoir la possibilité de faire évoluer dynamiquement le nombre de sommets et ainsi ne pas être obligé de connaître à priori la structure du graphe à implémenter. En effet, il suffit d’ajouter une nouvelle entrée au dictionnaire pour ajouter un nouveau sommet au graphe.

Le coût des opérations d’un dictionnaire d’adjacence est optimal. Ajouter un arc ou tester la présence d’un arc se fait à temps constant et parcourir les voisins d’un sommet donné se fait en un temps proportionnel au nombre de voisins.

Le seul intérêt de la matrice d’adjacence peut résider dans l’espace mémoire qu’elle occupe dans certains cas. Ainsi un graphe de 400 sommets contenant presque tous les arcs possibles occupe 10 fois moins de place quand il est représenté par une matrice d’adjacence que lorsqu’il est représenté par un dictionnaire d’adjacence.