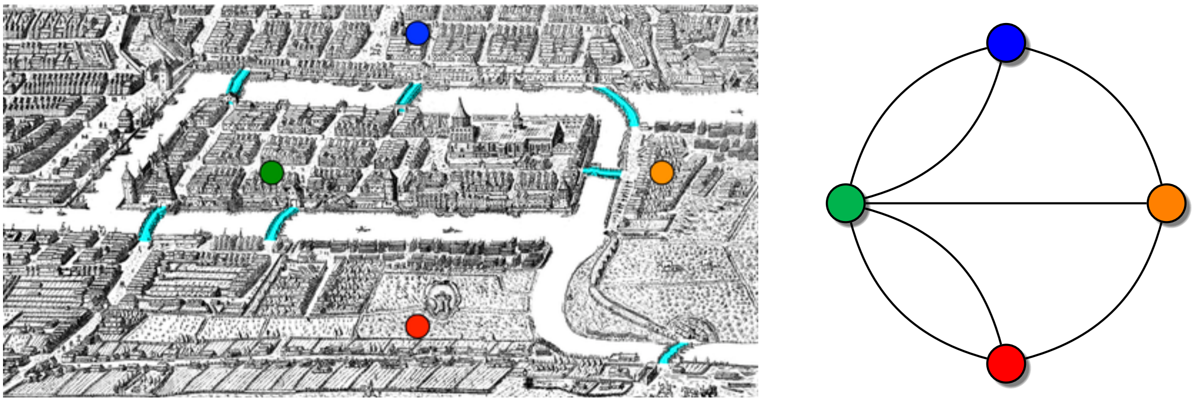
Graphes : définitions,

représentations et implémentations



1735 - Konigsberg. Euler pose les bases de la théorie des graphes.

Objectifs

A la fin de la séquence d'enseignement les élèves doivent :

- maitriser le vocabulaire *courant* associé aux graphes
- pourvoir représenter un graphe sous forme schématique, de listes d'adjacence ou de matrices d'adjacence
- être capable de traduire une graphe en langage **Python** à l'aide des listes et/ou de dictionnaires

Table des matières

1 Vocabulaire et définitions

1.1 D'un point de vue mathématique

Un graphe G est constitué de deux éléments :

- un ensemble de points noté $S = \{s_0, s_i, \ldots, s_{p-1}\}$ (ou V pour *vertex* - *vertices* en anglais)
- un ensemble noté $A = \{a_0, a_1, \ldots, a_{n-1}\}$ (ou E pour *edge* en anglais) de couples (s_i, s_j) ou paires $\{s_i, s_j\}$ avec s_i et s_j des éléments de S.

Une paire $\{s_i, s_j\}$ décrit une relation symétrique entre s_i et s_j. En revanche, un couple (s_i, s_j) traduit une relation de s_i vers s_j sans qu'il existe pour autant une relation de s_j vers s_i.

1.2 Vocabulaire et définitions associés aux graphes

- Un graphe G(S, A) est constitué d'un ensemble S de sommets (points ou nœuds) et d'un ensemble A d'arêtes (couples, paires, lignes, liens. . .).
- L'ordre n d'un graphe G(S, A) est le nombre de ses sommets. On note l'ordre $p = |S| = \#S$.
- Un graphe est orienté s'il existe un sens de parcours des arêtes. Les arêtes ont alors un sens et s'appellent alors arc.
- Un graphe est dit vide si l'ensemble A de ses arêtes est vide.
- Un graphe est simple si deux sommets quelconques sont reliés par au plus une arête et qu'aucune arête possède le même sommet à ses extrémités.
- Un multigraphe est un graphe pour lequel deux sommets sont reliés entre eux par plusieurs arêtes et/ou une arête possède le même sommet à ses extrémités.
- Un graphe est dit planaire (sinon non planaire) s'il est possible de le représenter dans le plan sans que ses arêtes se croisent.
- Un graphe est dit connexe si quelque soit un de ses sommets, il est possible d'atteindre tous les autres sommets via les arêtes.
  REMARQUE : un graphe non connexe se décompose en sous-graphe connexes.
- Un graphe est dit complet si chacun de ses sommets est reliés à tous les autres.
- Un graphe est dit pondéré si un nombre ou un poids est associé à chaque arête.
- Un graphe est dit étiqueté si un texte ou un étiquette est associé à chaque arête.

1.3 Vocabulaire et définitions liés aux sommets et aux arêtes

1.3.1 Dans un graphe quelconque

- Deux sommets sont dits adjacents s'ils sont reliés par une même arête. Ces deux sommets sont alors appelés extrémités de l'arête et sont définis comme voisins.
- Le nombre de voisins d'un sommet s_i est son degré que l'on note $d(s_i)$.
- Une arête forme une boucle si ses extrémités sont identiques (ex : $\{s_i, s_i\}$).
- Le degré d'un graphe est le degré maximum de tous ses sommets.

1.3.2 Dans un graphe orienté

- les arêtes sont appelées arcs

- un arc a une extrémité initiale (sommet de départ, origine ou prédécesseur) et une extrémité finale ou extrémité terminale (sommet d'arrivée, extrémité ou successeur).
- un sommet s_i a un degré entrant (nombre d'arcs dont il est l'extrémité finale) noté $d_-(s_i)$ aussi appelé demi-degré intérieur et un degré sortant (nombre d'arcs dont il est l'extrémité initiale) noté $d_+(s_i)$ aussi appelé demi-degré extérieur.

1.4 Vocabulaire et définitions liés au parcours d'un graphe

- Dans un graphe non orienté, on appelle chaine, une séquence $(s_0, s_1, \dots, s_c)$ où deux sommets consécutifs sont adjacents. Le sommet de départ est s_0 et le sommet d'arrivé s_c.
- Dans un graphe orienté, on parle de chemin plutôt que de chaine et on note $\chi(A,B)$, le chemin de A vers B.
- Un chemin peut être donné par la séquence des sommets ou par la séquence des arcs empruntées $(a_0, a_1, \dots, a_{c-1})$.
- La longueur d'une chaine (ou d'un chemin) est le nombre d'arêtes utilisées pour aller du sommet de départ s_0 au sommet d'arrivé s_c.
- La distance entre deux sommets est la longueur minimale d'une chaine (ou d'un chemin) reliant ces deux sommets.
- On appelle diamètre d'un graphe, la plus longue distance entre deux sommets. **Remarque :** s'il n'existe pas de chaine ou de chemin reliant ces deux sommets, la distance entre ces sommets est infinie.

1.5 Vocabulaire et définitions liés aux chaines

- Une chaine est dite simple si chaque arête du graphe y apparait au plus une fois.
- Une chaine est dite fermée si le sommet de départ est le sommet de fin.
- Un cycle est une chaîne fermée simple.
- Un cycle eulérien d'un graphe est un cycle passant une et une seule fois par chacune des arêtes du graphe.
- Un graphe est dit eulérien s'il possède un cycle eulérien.
- Une chaine eulérienne d'un graphe est une chaine passant une et une seule fois par chacune des arêtes du graphe.
- Un cycle hamiltonien d'un graphe est un cycle passant une et une seule fois par chacun des sommets du graphe.
- Un graphe est dit hamiltonien s'il possède un cycle hamiltonien.
- Une chaine hamiltonienne d'un graphe est une chaine passant une et une seule fois par chacun des sommets du graphe.

Pour un graphe G ayant n arêtes, p sommets et c composantes connexes, on définit le nombre cyclomatique $\nu(G)$ du graphe par : $\nu(G) = n - p + c$. Le nombre cyclomatique représente le nombre de cycles indépendants.

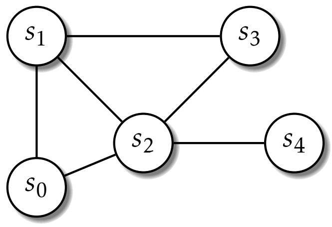
- Un arbre est un graphe connexe de nombre cyclomatique nul.
- Une forêt est graphe non connexe composé d'arbres.

2 Représentations des graphes

2.1 Représentations graphiques

Le plus simple pour représenter un graphe est de le dessiner sous forme de schéma.

Exemple : G(S, A), graphe non orienté, connexe, d'ordre 5 et de degré 4.

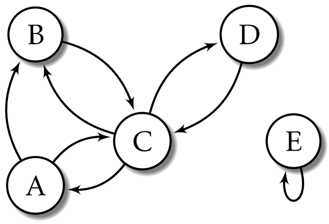


$$\begin{aligned} S &= \{s_0, s_1, s_2, s_3, s_4\} \\ A &= \{\{s_0, s_1\}, \{s_0, s_2\}, \{s_1, s_2\}, \{s_1, s_3\}, \{s_2, s_3\}\{s_2, s_4\}\} \\ & \quad d(s_2) = 4 \quad \text{et} \quad d(s_4) = 1 \\ \nu(G) &= 6 - 5 + 1 = 2 \end{aligned}$$

Les voisins du sommet s_2 sont s_0, s_1, s_3 et s_4 et l'unique voisin du sommet s_4 est s_2.

Exemple :
G(S, A), graphe orienté non connexe, d'ordre 5 et de degré 6.

$\nu(G) = 8 - 5 + 2 = 5$

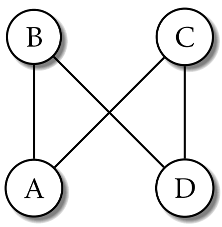


$$\begin{aligned} S &= \{A, B, C, D, E\} \\ A &= \{(A, B), (A, C), (C, A), (B, C), (C, B), (C, D), (D, C), (E, E)\} \\ & \quad d_+(B) = 1 \quad ; \quad d_-(B) = 2 \quad \text{et} \quad d(B) = 3 \end{aligned}$$

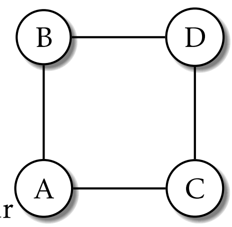
(D, C, B) est une chaine et (A, B, C, A) est une chaine fermée.

(B, A, C) n'est pas une chaine (et pas non plus un chemin). (A, B, C, D) est une chaine de longueur 3. A est à une distance de D de 2 (via la chaine (A, C, D)).

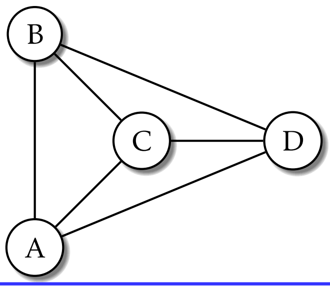
Graphe planaire
G({A, B, C, D}, {{A, B}, {B, D}, {D, C}, {C, A}})



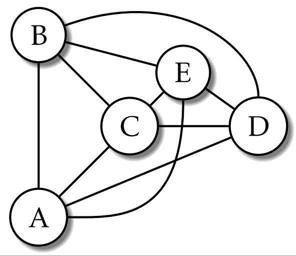
car



Graphe complet planaire :



Graphe complet non planaire :



2.2 Représentations par adjacence

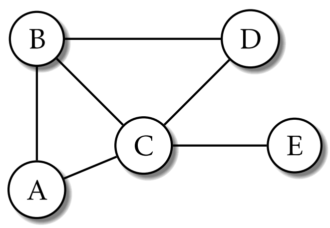
Une façon de représenter un graphe est de définir pour chaque nœud :

- soit l'ensemble de ses voisins,
- soit l'ensemble de ses successeurs,
- soit l'ensemble de ses prédécesseurs,
- soit l'ensemble des arêtes dont il est extrémité,
- soit l'ensemble des arcs incidents vers l'extérieur (sortant)
- soit l'ensemble des arcs incidents vers l'intérieur (entrant).

2.2.1 Listes d'adjacences

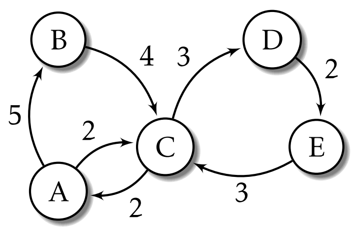
Donner pour chaque sommet la liste de ses voisins, c'est ce qu'on appelle donner les listes d'adjacence.

L'ordre d'écriture n'a pas d'importance et nous verrons par la suite l'intérêt des dictionnaires dans l'étude des graphes.



A	:	B, C
B	:	A, C, D
C	:	A, B, D, E
D	:	B, C
E	:	C

A	:	(B, 5), (C, 2)
B	:	(C,4)
C	:	(A,2), (D, 3)
D	:	(E,2)
E	:	(C,3)



Dans le cas d'un graphe orienté, on peut donner, pour chaque nœud, la liste des successeurs.

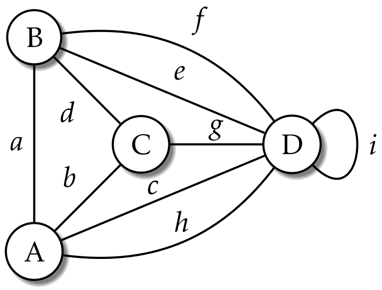
Si le graphe est pondéré, on inscrit le poids à côté du sommet ou de l'arête.

2.2.2 Représentation par tableaux et matrices

Pour un graphe G(S, A), les liens entre les sommets s_i ou entre sommets s_i et arêtes a_k peuvent se traduire à l'aide de tableaux à deux entrées.

2.2.2.1 Tableau ou matrice d'incidence d'un graphe non-orienté

La matrice d'incidence $\mathbb{M}_G$ de G est une matrice $p \times n$ où le coefficient $\mathbb{M}_{Gik}$ représente le nombre d'incidences entre le sommet s_i et l'arête a_k.



	a	b	c	d	e	f	g	h	i
A	1	1	1	0	0	0	0	1	0
B	1	0	0	1	1	1	0	0	0
C	0	1	0	1	0	0	1	0	0
D	0	0	1	0	1	1	1	1	2

$$\mathbb{M}_G = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 2 \end{pmatrix}$$

2.2.2.2 Tableau ou matrice d'adjacence d'un graphe non-orienté non pondéré

Comme le nombre d'arêtes est souvent bien supérieur au nombre de sommets, on préfère utiliser un tableau ou matrice d'adjacence $\mathbb{A}_G$ de G, matrice carrée $p \times p$ où le coefficient $\mathbb{A}_{Gij}$ représente le nombre d'arêtes entre le sommet s_i et le sommet s_j. Le graphe étant non orienté, le tableau est symétrique.

Exemple : en reprenant le graphe de la partie 2.2.2.1.

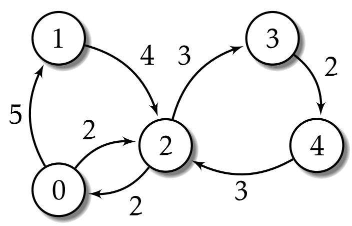
	A	B	C	D
A	0	1	1	2
B	1	0	1	2
C	1	1	0	1
D	2	2	1	2

$$\mathbb{A}_G = \begin{pmatrix} 0 & 1 & 1 & 2 \\ 1 & 0 & 1 & 2 \\ 1 & 1 & 0 & 1 \\ 2 & 2 & 1 & 2 \end{pmatrix}$$

2.2.2.3 Tableau ou matrice d'adjacence d'un graphe orienté et pondéré

Dans le cas d'un graphe orienté, la matrice d'adjacence n'est pas forcément symétrique. On place sur le coefficient $\mathbb{A}_{Gij}$ un 1 (ou True) s'il existe un arc de s_i vers s_j et un 0 (ou False) sinon.

Si le graphe est pondéré, on remplace les 1 par le poids de l'arc.



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

3 Implémentation des graphes

D'un point de vue informatique, plusieurs structures permettent de représenter les graphes. Les objets **Python** `list`, `array` et `dict` sont particulièrement adaptés.

3.1 Implémentation à partir d'une matrice d'adjacence

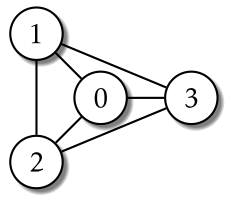
Si on pose `G = [[0, 1, 1, 1], [1, 0, 1, 1], [1, 1, 0, 1], [1, 1, 1, 0]]`, on perçoit que `G` est un graphe complet à 4 sommets.

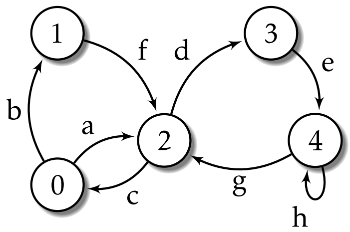
```
>>> T = np.array(G)
>>> T
array([[0, 1, 1, 1],
       [1, 0, 1, 1],
       [1, 1, 0, 1],
       [1, 1, 1, 0]])
```

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

`G[i]` donne la liste des relations entre le sommet `i` et les autres sommets.

Si `G[i][j] == 0`, il n'y pas d'arête entre les sommets `i` et `j`.






```
>>> T
array([['0', 'b', 'a', '0', '0'],
       ['0', '0', 'f', '0', '0'],
       ['c', '0', '0', 'd', '0'],
       ['0', '0', '0', '0', 'e'],
       ['0', '0', 'g', '0', 'h']], dtype='<U11')
```

L'élément `T[i, j]` traduit la présence d'arc entre les sommets s_i et s_j en y référant son étiquette.

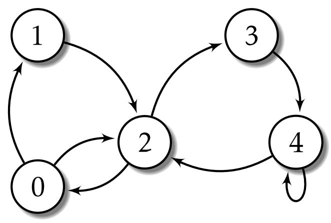
`T[i]` donne la liste des arcs incidents vers l'extérieur du sommet s_i et `T[:, j]` donne la listes des arcs incidents vers l'intérieur du sommet s_j.

3.2 Implémentation à partir de listes d'adjacence

Dans la partie 3.1, l'utilisation de tableau oblige à stocker des 0 quand il n'existe pas d'arête entre deux sommets.

Avec une structure par liste d'adjacence, on ne stocke, pour chaque sommet, que ses voisins :

```
G = [[1, 2], [2], [3, 0], [4], [2, 4]]
```



Ainsi `G[i]` donne la liste des voisins du sommet `i`.

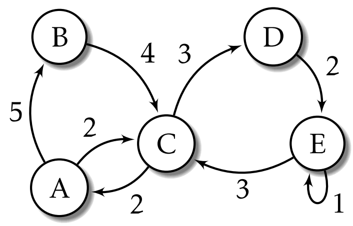
Cependant, les sommets ne sont pas toujours des entiers. On peut alors représenter un graphe `G` par une liste de listes pour laquelle l'élément `i` est constitué du nom du sommet `i` et de la liste des voisins du sommet `i` :

```
G = [[0, [1, 2]], [1, [2]], [2, [3, 0]], [3, [4]], [4, [2, 4]]]
```

3.3 Implémentation à partir de dictionnaires

Étant donné l'absence de relation d'ordre entre les nœuds et entre les voisins, l'utilisation de dictionnaires s'avère plus adaptée. Les clés du dictionnaires sont les noms des sommets et les valeurs des clés, la liste des voisins de la clé.

```
G = {0 : [1, 2], 1 : [2], 2 : [3, 0], 3 : [4], 4, :[2, 4]}
```



Dans le cas d'un graphe pondéré, il faut ajouter à chaque voisin le poids de l'arc qui y conduit. On se tourne alors vers un dictionnaire de ...dictionnaires !

```
G ={'A' : {'B' : 5, 'C' : 2}, 'B' : {C : 4}, 'C' : {'A' : 2, 'D' : 3},
      'D' : {'E' : 2}, 'E' : {'C' : 3, 'E' : 1}}
```