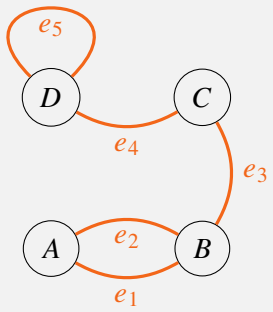
Graphes

Définition 1 : Un graphe est un ensemble de points (sommets) reliés par des traits (arêtes).

▪ **Exemple 1 :** On considère le graphe $\mathcal{G}$ dont la représentation graphique est la suivante.



Ce graphe possède 4 sommets et 5 arêtes.

Cette approche visuelle, bien qu'intuitive, montre vite ses limites pour manipuler les graphes avec toute la rigueur mathématique requise.

1 Différents types de graphes

1.1 Graphe non orienté

Définitions

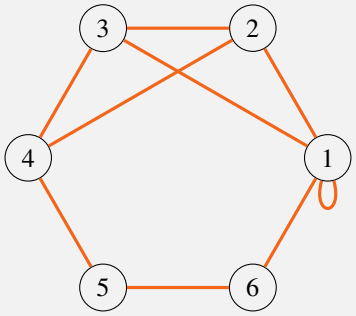
Définition 2 — Graphe non orienté : Un graphe non orienté $\mathcal{G}$ est un objet mathématique défini par un couple $(\mathcal{S}, \mathcal{A})$ où :

- $\mathcal{S}$ est un ensemble (non vide) désignant les **sommets** du graphe ;
- $\mathcal{A}$ est une **famille** (ou collection) de parties de $\mathcal{S}$ à un ou deux éléments, désignant les **arêtes**.
  - ► Une arête constituée de deux éléments distincts $\{x, y\}$ relie le sommet x au sommet y.
  - ► Une arête constituée d'un seul élément $\{x\}$ relie le sommet x à lui-même : c'est une **boucle**.
  - ► Comme $\mathcal{A}$ est une famille, une même paire de sommets peut y figurer plusieurs fois : le graphe peut donc comporter des **arêtes multiples**.

On appelle **ordre** du graphe $\mathcal{G}$ le nombre de sommets de ce graphe.

On dit que le graphe $\mathcal{G}$ est **simple** s'il ne contient ni boucle ni arête multiple.

▪ **Exemple 2 :** On considère le graphe non orienté $\mathcal{G}$ ci-dessous.



Ce graphe possède 6 sommets (il est donc d'ordre 6), aucune arête multiple, et comporte une boucle sur le sommet 1.

Donnons la formalisation de ce graphe $\mathcal{G} = (\mathcal{S}, \mathcal{A})$:

- **Les sommets :** L'ensemble $\mathcal{S}$ est constitué des six sommets distincts :

$$\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$$

- **Les arêtes :** L'ensemble $\mathcal{A}$ contient les parties à un ou deux éléments correspondant aux connexions tracées :

$$\mathcal{A} = \Big\{\{1,2\}, \{2,3\}, \{3,4\}, \{4,5\}, \{5,6\}, \{6,1\}, \{1,3\}, \{2,4\}, \{1\}\Big\}$$

Ce graphe n'est pas simple puisqu'il comporte une boucle.

Définition 3 : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe non orienté. Soit x et y deux sommets de $\mathcal{G}$.

- x et y sont **adjacents** ou **voisins** si $\{x, y\} \in \mathcal{A}$: autrement dit, s'il existe une arête reliant x et y ;
- x est **isolé** s'il n'est relié à aucun autre sommet.

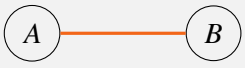
▪ **Exemple 3 :** Dans le graphe précédent, les sommets 1 et 2 sont adjacents, mais pas les sommets 1 et 4. Il n'y a pas de sommet isolé.

Définition 4 — Graphe complet : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe.

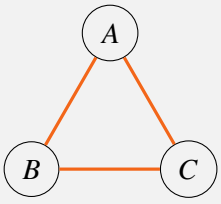
On dit que $\mathcal{G}$ est un graphe complet si $\mathcal{G}$ est un graphe simple dont tous les sommets sont adjacents deux à deux.

▪ **Exemple 4 :** Les graphes suivants sont complets (chaque sommet est relié à tous les autres). On les note traditionnellement K_n, où n est l'ordre du graphe.

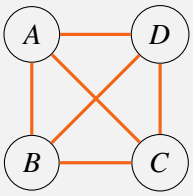
Ordre 2 (K_2)



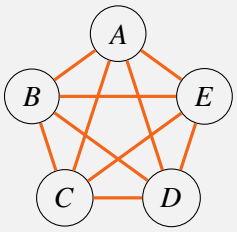
Ordre 3 (K_3)



Ordre 4 (K_4)



Ordre 5 (K_5)



Propriété 1 : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe simple non orienté d'ordre n.

Le graphe $\mathcal{G}$ est complet si et seulement si le nombre d'éléments de $\mathcal{A}$ est $\dfrac{n(n-1)}{2}$.

Démonstration 1 : Le nombre de paires d'éléments distincts de $\mathcal{S}$ vaut $\binom{n}{2}$ soit $\dfrac{n(n-1)}{2}$. Le graphe est complet si et seulement si toute paire de sommet est relié par une arête. ▫

▪ **Exemple 5 :** Un graphe simple non orienté ayant 6 sommets et 15 arêtes est complet.

En effet, $\frac{6\times(6-1)}{2} = \frac{6\times 5}{2} = 15$.

Degré d'un sommet

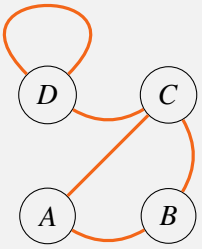
Définition 5 : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe non orienté. Soit x un sommet de $\mathcal{G}$.

On appelle degré de x, noté $\deg(x)$, le nombre d'arêtes dont x est une extrémité, **chaque boucle étant comptée deux fois**.

▪ **Exemple 6 :**

Dans le graphe suivant...

- Le sommet A est de degré 2
- Le sommet B est de degré 2
- Le sommet C est de degré 3
- Le sommet D est de degré 3



Propriété 2 — Formule d'Euler : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe non orienté. Alors

$$\sum_{x\in\mathcal{S}} \deg(x) = 2|\mathcal{A}|$$

La somme des degrés des sommets de $\mathcal{G}$ est égale à deux fois son nombre d'arêtes.

En particulier, la somme des degrés d'un graphe est un nombre pair.

Démonstration 2 : Lorsque l'on somme les degrés des sommets, chaque arête est en effet comptée deux fois (une pour chaque extrémité). □

▪ **Exemple 7 :** Est-il possible d'établir un réseau filaire de 5 ordinateurs en faisant en sorte que chaque ordinateur soit exactement relié à trois autres ?

Supposons que cela soit possible. Nous construisons alors un graphe $\mathcal{G}$ dont l'ensemble des sommets correspond aux cinq ordinateurs, et où deux sommets sont reliés si les deux ordinateurs auxquels ils correspondent sont liés. D'après l'énoncé, tous les sommets de ce graphe sont de degré 3. La somme des degrés de ce graphe vaut donc 15.

Or, la somme des degrés des sommets d'un graphe est un entier pair : une telle construction est donc impossible.

Remarque : il est possible de retrouver la caractérisation des graphes complets à l'aide de cette formule. En effet, si un graphe $\mathcal{G}$ d'ordre n est complet, alors tous ses sommets sont de degré $n-1$.

Or, $\sum_{x\in\mathcal{S}} (n-1) = 2|\mathcal{A}|$ et donc $|\mathcal{A}| = \frac{n(n-1)}{2}$.

Propriété 3 — Lemme des poignées de main : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe non orienté.

Alors $\mathcal{G}$ possède un nombre pair de sommets de degré impair.

Démonstration 3 : Notons D_i la somme des degrés impairs et D_p la somme des degrés pairs.

Alors, d'après la proposition précédente, $D_i + D_p$, qui est la somme de tous les degrés, est un nombre pair. Soit donc $p \in \mathbb{N}$ tel que $D_i + D_p = 2p$.

On a donc $D_i = 2p - D_p$. Or, D_p est une somme de nombres pairs, c'est donc également un nombre pair. Ainsi, $2p - D_p$ est pair.

Or, D_i est une somme de nombres impairs : cette somme contient donc un nombre pair de nombres. □

▪ **Exemple 8 :** En reprenant le problème précédent, on remarque que le problème se ramène à la construction d'un graphe à 5 sommets de degré 3. Il y aurait donc un nombre impair de sommets de degrés impairs, ce qui est impossible.

1.2 Graphe orienté

Définitions

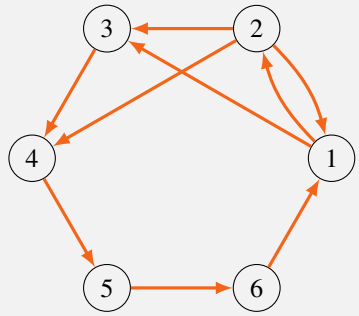
Les graphes précédents étaient non orientés puisqu'une paire de sommets n'admet pas d'ordre. La paire $\{A, B\}$ est par exemple la même que la paire $\{B, A\}$. Si l'on souhaite représenter un graphe orienté, il suffit de remplacer le terme de paires par celui de couples.

Définition 6 — Graphe orienté : Un graphe orienté $\mathcal{G}$ est un objet mathématique défini par un couple $(\mathcal{S}, \mathcal{A})$ où :

- $\mathcal{S}$ est un ensemble désignant les sommets du graphe $\mathcal{G}$;
- $\mathcal{A}$ est une famille de **couples** de $\mathcal{S}$ à un ou deux éléments, désignant les **arcs** du graphe $\mathcal{G}$.
  - ► Un arc constitué de deux éléments distincts (x, y) relie le sommet x au sommet y.
    Les sommets x et y sont appelés extrémités de l'arc (x, y).
  - ► Un arc constituée d'un seul élément (x, x) relie le sommet x à lui-même : c'est une **boucle**.

On dit que le graphe $\mathcal{G}$ est simple s'il ne contient ni boucle ni arête multiple.

▪ **Exemple 9 :** On considère le graphe orienté $\mathcal{G}$ ci-dessous.



Ce graphe orienté d'ordre 6 ne possède aucune boucle. On remarque que les sommets 1 et 2 sont reliés dans les deux sens.

Donnons la formalisation de ce graphe $\mathcal{G} = (\mathcal{S}, \mathcal{A})$:

- **Les sommets :**

$$\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$$

- **Les arcs :** L'ensemble $\mathcal{A}$ n'est plus constitué de paires, mais de **couples** (car l'ordre importe). C'est un sous-ensemble du produit cartésien $\mathcal{S} \times \mathcal{S}$:

$$\mathcal{A} = \Big\{(1,2), (2,1), (2,3), (3,4), (4,5), (5,6), (6,1), (1,3), (2,4)\Big\}$$

Remarque importante : Bien qu'il y ait deux flèches entre les sommets 1 et 2, ce graphe est **simple**. En effet, les couples $(1, 2)$ et $(2, 1)$ sont distincts. Il n'y a donc pas d'arcs multiples de même sens reliant les mêmes sommets.

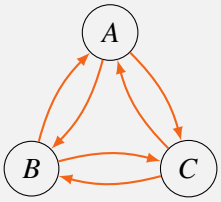
Définition 7 — Graphe complet : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe orienté.

On dit que $\mathcal{G}$ est un graphe complet si $\mathcal{G}$ est un graphe simple pour lequel chaque sommet est relié à tous les autres sommets.

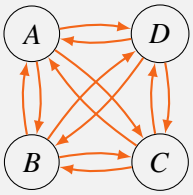
▪ **Exemple 10 :** Les graphes orientés suivants sont complets : chaque sommet est relié à tous les autres par un arc dans les deux sens (on parle de graphe orienté symétrique complet). On les note traditionnellement K_n^*.

Ordre 2 (K_2^*)

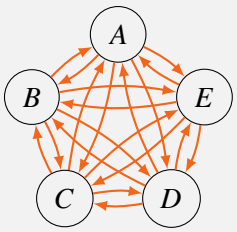
Ordre 3 (K_3^*)



Ordre 4 (K_4^*)



Ordre 5 (K_5^*)



Propriété 4 : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe simple orienté d'ordre n.
Le graphe $\mathcal{G}$ est complet si et seulement si le nombre d'éléments de $\mathcal{A}$ est $n(n-1)$.

Démonstration 4 : Le nombre de couples d'éléments distincts de $\mathcal{S}$ vaut $n(n-1)$. Le graphe est complet si et seulement si tout couple de sommet est relié par une arête. □

▪ **Exemple 11 :** Un graphe simple orienté ayant 7 sommets est complet si et seulement s'il a 42 arêtes.

Degré d'un sommet

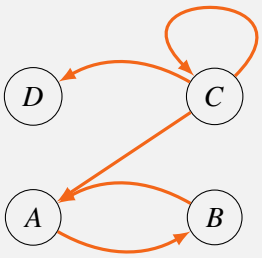
Définition 8 — Degré d'un sommet dans un graphe orienté : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe orienté. Soit x un sommet de $\mathcal{S}$.

- On appelle degré sortant de x, noté $\deg_+(x)$, le nombre d'arêtes dont x est l'extrémité de départ.
- On appelle degré entrant de x, noté $\deg_-(x)$, le nombre d'arêtes dont x est l'extrémité d'arrivée.
- On appelle degré de x, noté $\deg(x)$, la somme de son degré entrant et de son degré sortant.

▪ **Exemple 12 :**

Dans le graphe suivant, on a...

Sommet	A	B	C	D
Degré sortant	1	1	3	0
Degré entrant	2	1	1	1
Degré	3	2	4	1



Propriété 5 — Formule d'Euler : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe orienté. Alors

$$\sum_{x \in \mathcal{S}} \deg_+(x) = \sum_{x \in \mathcal{S}} \deg_-(x) = |\mathcal{A}|$$

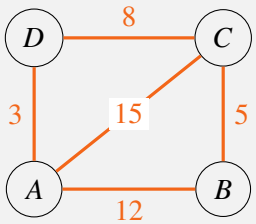
1.3 Graphes pondérés (ou valués)

Définition 9 — Graphe pondéré : Un graphe (orienté ou non orienté) est dit **pondéré** (ou valué) si l'on associe à chaque arête (ou arc) un nombre réel. Ce nombre est appelé le **poids** de l'arête.

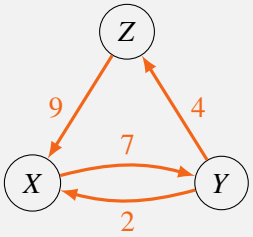
Selon le contexte modélisé par le graphe, ce poids peut représenter une distance kilométrique, un coût financier, une durée, ou encore une probabilité.

▪ **Exemple 13 :** Voici deux exemples de graphes pondérés. Pour faciliter la lecture, le poids de chaque arête (ou arc) est inscrit à côté de celle-ci.

Graphe non orienté pondéré

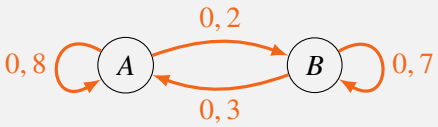


Graphe orienté pondéré



▪ **Exemple 14 :** Les graphes orientés pondérés sont utilisés pour représenter l'évolution d'un système aléatoire au cours du temps. On parle alors de **graphe probabiliste**.

Étudions le marché d'une plateforme de streaming vidéo. Chaque mois, un client potentiel peut être dans l'un de ces deux états : L'état A : « Être abonné à la plateforme » et l'état B : « Ne pas (ou ne plus) être abonné ».



- L'arc de A vers B indique qu'un abonné a une probabilité de $0,2$ de se désabonner le mois suivant.
- La boucle sur A indique qu'un abonné a une probabilité de $0,8$ de rester abonné.

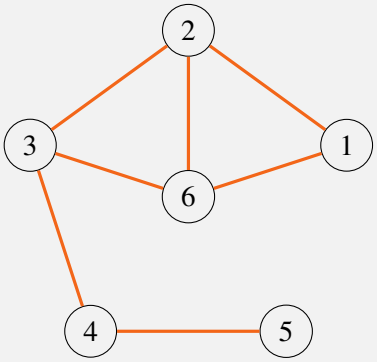
2 Parcours d'un graphe

2.1 Chaîne dans un graphe non orienté

Définition 10 — Chaîne dans un graphe non orienté : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe non orienté. Soit x et y deux sommets de ce graphe.

- Une **chaîne** reliant un sommet x à un sommet y est une suite finie d'au moins deux sommets $(x_0, x_1, \ldots, x_k)$ avec $k \geqslant 1$, telle que le premier sommet est $x_0 = x$, le dernier est $x_k = y$, et où chaque sommet est adjacent à son successeur ;
- La **longueur** d'une chaîne est égale au nombre d'arêtes **empruntées** (ou parcourues) lors de ce trajet. Une chaîne constituée de $k+1$ sommets est donc de longueur k.
- Une chaîne est élémentaire si elle ne contient pas deux fois un même sommet ;
- Une chaîne est **fermée** lorsque le sommet initial et le sommet final sont identiques :
- Une chaîne est **simple** lorsque toute arête qui la constitue n'est empruntée qu'une seule fois ;
- Un **cycle** est une chaîne simple fermée. Autrement dit, c'est une chaîne dans laquelle chaque arête n'est parcourue qu'une fois et dont le sommet de départ est identique à celui d'arrivée.

▪ **Exemple 15 :** Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ le graphe non orienté d'ordre 6 représenté ci-dessous.



- **Chaîne de 1 à 4 :**
  - *Exemple :* $\mathcal{C}_1 = (1, 6, 2, 3, 4)$ est une chaîne reliant 1 à 4.
  - *Contre-exemple :* $\mathcal{C}_2 = (1, 6, 4)$ n'est pas une chaîne car les sommets 6 et 4 ne sont pas adjacents.
- **Longueur d'une chaîne :**
  - La chaîne $\mathcal{C}_1$ ci-dessus est constituée de 5 sommets. Elle emprunte 4 arêtes, sa longueur est donc de **4**.
- **Chaîne élémentaire :**
  - *Exemple :* $\mathcal{C}_1 = (1, 6, 2, 3, 4)$ est élémentaire (tous les sommets sont distincts).
  - *Contre-exemple :* $\mathcal{C}_3 = (1, 2, 6, 1, 6)$ n'est pas élémentaire car le sommet 1 apparaît deux fois.
- **Chaîne fermée :**
  - *Exemple :* $\mathcal{C}_4 = (1, 2, 6, 1)$ est fermée (commence et finit par 1).
  - *Contre-exemple :* $\mathcal{C}_1$ n'est pas fermée ($1 \neq 4$).
- **Chaîne simple :**
  - *Exemple :* $\mathcal{C}_4 = (1, 2, 6, 1)$ emprunte les arêtes $\{1, 2\}, \{2, 6\}, \{6, 1\}$. C'est une chaîne simple.
  - *Contre-exemple :* $\mathcal{C}_3 = (1, 2, 6, 1, 6)$ n'est pas simple car l'arête $\{6, 1\}$ est empruntée deux fois.
- **Cycle :**
  - *Exemple :* $\mathcal{C}_4 = (1, 2, 6, 1)$ est une chaîne simple fermée : c'est un cycle.
  - *Contre-exemple :* La chaîne $\mathcal{C}_5 = (1, 2, 6, 2, 1)$ est bien une chaîne fermée, mais elle n'est pas simple (l'arête $\{2, 6\}$ est empruntée deux fois à l'aller et au retour). Ce n'est donc pas un cycle.

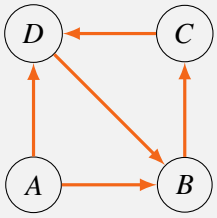
2.2 Chemin dans un graphe orienté

Le vocabulaire des graphes orientés est le strict pendant de celui des graphes non orientés, mais on remplace les mots « chaîne » et « cycle » par « chemin » et « circuit » pour insister sur le fait que l'on doit **respecter l'orientation du graphe**.

Définition 11 — Chemin et circuit : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe orienté.

- Un **chemin** allant d'un sommet x à un sommet y est une suite finie d'au moins deux sommets $(x_0, x_1, \ldots, x_k)$ avec $k \geqslant 1$, telle que $x_0 = x$, $x_k = y$, et où pour chaque étape, le couple (x_i, x_{i+1}) est un arc de $\mathcal{A}$ (il y a une flèche de x_i vers x_{i+1}).
- La **longueur** d'un chemin est le nombre d'arcs empruntés (soit k).
- Un chemin est **élémentaire** si tous ses sommets sont distincts.
- Un chemin est **simple** si tous ses arcs sont distincts.
- Un **circuit** est un chemin simple fermé (le sommet de départ est le même que le sommet d'arrivée).

▪ **Exemple 16 :** Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ le graphe orienté d'ordre 4 représenté ci-dessous.



- **Chemin :** $\mathcal{C}_1 = (A, B, C, D)$ est un chemin de longueur 3 reliant A à D.
- **Contre-exemple :** (A, D, C) **n'est pas un chemin**. Les sommets D et C sont adjacents, mais la flèche va de C vers D. On ne peut pas remonter à contre-sens dans un graphe orienté !
- **Circuit :** $\mathcal{C}_2 = (B, C, D, B)$ est un circuit de longueur 3.

2.3 Connexité d'un graphe

Intuitivement, un graphe est connexe s'il est « d'un seul tenant ». Autrement dit, il ne possède pas de sommets ou de sous-groupes de sommets totalement isolés du reste du réseau.

Cas des graphes non orientés

Définition 12 — Graphe connexe : Un graphe non orienté est dit **connexe** si pour toute paire de sommets distincts (x, y), il existe au moins une chaîne reliant x à y.

▪ **Exemple 17 :** Le graphe ci-dessous **n'est pas connexe**. Il est constitué de deux composantes isolées l'une de l'autre. Il n'existe par exemple aucune chaîne permettant de relier le sommet 1 au sommet 4.

Cas des graphes orientés

Dans un graphe orienté, les flèches peuvent créer des « voies sans issue » ou des sommets inatteignables. On distingue donc deux niveaux de connexité.

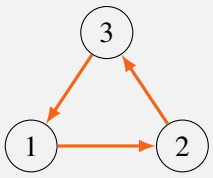
Définition 13 — Connexité forte et faible : Soit $\mathcal{G}$ un graphe orienté.

- $\mathcal{G}$ est dit **fortement connexe** si pour tous sommets distincts x et y, il existe un chemin de x vers y **ET** un chemin de y vers x.
- $\mathcal{G}$ est dit **faiblement connexe** si, en remplaçant tous ses arcs par des arêtes non orientées, le graphe non orienté obtenu est connexe.

Par abus de langage, lorsqu'un ouvrage parle d'un graphe orienté simplement « connexe », il sous-entend généralement une connexité faible. Toutefois, il est préférable de toujours prendre le soin de préciser l'adjectif approprié.

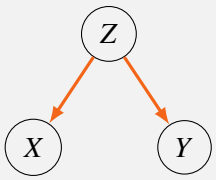
▪ **Exemple 18 :** On considère les deux graphes suivants.

Fortement connexe



On peut relier n'importe quel sommet à n'importe quel autre en suivant les flèches (ex : pour aller de 2 à 1, on fait le chemin $2 \to 3 \to 1$).

Faiblement connexe



Si l'on efface les flèches, le graphe est d'un seul tenant (connexe). Mais avec les flèches, il est **impossible** de partir de X ou de Y pour aller n'importe où.

2.4 Graphes eulériens et semi-eulériens

Dans cette partie, on s'intéresse au parcours des **arêtes** d'un graphe **non orienté connexe**. L'objectif est de trouver un trajet passant par chaque arête sans jamais lever le crayon.

Définition 14 — Chaîne et cycle eulériens : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe non orienté connexe.

- Une **chaîne eulérienne** est une chaîne qui emprunte **toutes les arêtes** du graphe $\mathcal{G}$ une et une seule fois.
- Un **cycle eulérien** est une chaîne eulérienne qui est fermée (le point de départ est le même que le point d'arrivée).

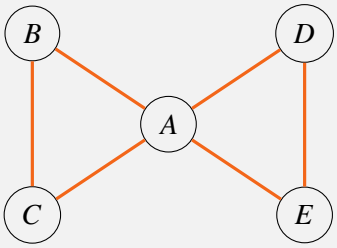
Définition 15 — Graphe eulérien et semi-eulérien :

- Un graphe est dit **eulérien** s'il possède au moins un cycle eulérien.
- Un graphe est dit **semi-eulérien** s'il possède au moins une chaîne eulérienne, mais aucun cycle eulérien.

Remarque : Le terme « eulérien » fait référence au mathématicien Leonhard Euler qui a résolu en 1736 le célèbre problème des sept ponts de Königsberg. Sur le papier, tracer une chaîne eulérienne revient à dessiner le graphe entier « sans lever le crayon » et sans repasser deux fois sur le même trait.

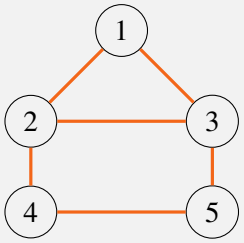
▪ **Exemple 19 :** Considérons les deux graphes suivants :

Graphe $\mathcal{G}_1$ (Le papillon)



Dans ce graphe, la chaîne (A, B, C, A, D, E, A) emprunte bien les 6 arêtes une unique fois. Puisque le point de départ et d'arrivée est identique (A), c'est un **cycle eulérien**.
Le graphe $\mathcal{G}_1$ est donc **eulérien**.

Graphe $\mathcal{G}_2$ (La maison)



La chaîne $(2, 1, 3, 2, 4, 5, 3)$ emprunte toutes les arêtes une seule fois. C'est donc une **chaîne eulérienne**. Cependant, elle n'est pas fermée. Il est impossible de créer un cycle eulérien.
Le graphe $\mathcal{G}_2$ est donc **semi-eulérien**.

Propriété 6 — Théorème d'Euler : Soit $\mathcal{G}$ un graphe non orienté connexe.

- **Caractérisation eulérienne :** Le graphe $\mathcal{G}$ est eulérien si et seulement si **tous ses sommets sont de degré pair**.
- **Caractérisation semi-eulérienne :** Le graphe $\mathcal{G}$ est semi-eulérien si et seulement si **exactement deux de ses sommets sont de degré impair**.

Remarque : Dans le cas d'un graphe semi-eulérien, les deux sommets de degré impair correspondent obligatoirement au **point de départ** et au **point d'arrivée** de la chaîne eulérienne.

▪ **Exemple 20 :** Vérifions ce théorème sur nos deux exemples précédents :

- Dans le graphe papillon $(\mathcal{G}_1)$, on a : $\deg(A) = 4$, et $\deg(B) = \deg(C) = \deg(D) = \deg(E) = 2$. Tous les sommets sont pairs, le graphe est bien eulérien.
- Dans le graphe maison $(\mathcal{G}_2)$, on a : $\deg(1) = \deg(4) = \deg(5) = 2$. Mais $\deg(2) = 3$ **et** $\deg(3) = 3$. Il y a exactement deux sommets de degré impair. Le graphe est bien semi-eulérien, et toute chaîne eulérienne reliera forcément 2 et 3.

3 Matrice d'adjacence

Pour manipuler un graphe de manière calculatoire (et notamment le faire traiter par un ordinateur), on utilise une matrice carrée qui répertorie toutes ses connexions. C'est le passage fondamental de la géométrie à l'algèbre.

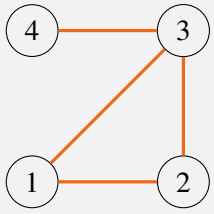
3.1 Définitions et premiers exemples

Définition 16 — Matrice d'adjacence d'un graphe non orienté : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe non orienté d'ordre n, dont les sommets sont numérotés de 1 à n.

La **matrice d'adjacence** de $\mathcal{G}$ est la matrice carrée $M = (m_{i,j})_{1 \leqslant i,j \leqslant n}$ de taille $n \times n$ définie par :

$$m_{i,j} = \text{nombre d'arêtes reliant le sommet } i \text{ et le sommet } j$$

▪ **Exemple 21 :** Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ le graphe non orienté d'ordre 4 représenté ci-dessous.



La matrice d'adjacence M associée à ce graphe est :

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

On remarque par exemple que $m_{1,3} = 1$ car l'arête $\{1, 3\}$ existe, mais $m_{1,4} = 0$ car 1 et 4 ne sont pas reliés.

Propriété 7 : Pour un graphe non orienté, la matrice d'adjacence est toujours **symétrique** ($M = {}^tM$).

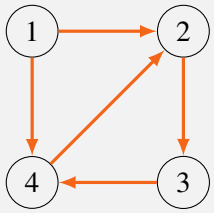
Démonstration 5 : En effet, l'arête $\{i, j\}$ n'étant pas ordonnée, le sommet i est adjacent à j si et seulement si j est adjacent à i. On a donc toujours $m_{i,j} = m_{j,i}$. □

Définition 17 — Matrice d'adjacence d'un graphe orienté : Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ un graphe orienté d'ordre n.

La **matrice d'adjacence** de $\mathcal{G}$ est la matrice carrée $M = (m_{i,j})_{1 \leqslant i,j \leqslant n}$ de taille $n \times n$ définie par :

$$m_{i,j} = \text{nombre d'arcs reliant le sommet } i \text{ au sommet } j \textbf{ dans ce sens}.$$

▪ **Exemple 22 :** Soit $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ le graphe orienté d'ordre 4 représenté ci-dessous.



La matrice d'adjacence M associée à ce graphe orienté d'ordre 4 est :

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

On observe que $m_{1,2} = 1$ (car il y a un arc de 1 vers 2), mais que $m_{2,1} = 0$ (car il n'y a pas d'arc de 2 vers 1). La matrice n'est donc **pas symétrique**.

3.2 Propriétés repérables sur la matrice

L'observation des coefficients de la matrice M permet de déduire immédiatement la nature géométrique du graphe :

- **Détection des boucles :** La présence d'un 1 sur la diagonale principale (un coefficient $m_{i,i} = 1$) indique l'existence d'une boucle sur le sommet i.
- **Graphe simple :** Un graphe est simple si sa matrice ne contient que des 0 et des 1, et si sa **diagonale principale ne contient que des 0** (aucune boucle).
- **Graphe complet (K_n) :** Sa matrice ne contient que des 1, sauf sur la diagonale principale qui ne contient que des 0.

3.3 Puissance de la matrice et dénombrement de chemins

C'est la propriété fondamentale de ce chapitre. Elle justifie à elle seule l'utilisation des matrices en théorie des graphes.

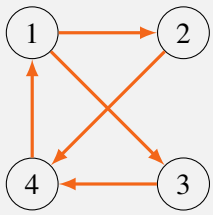
Propriété 8 — Théorème fondamental : Soit M la matrice d'adjacence d'un graphe (orienté ou non) d'ordre n. Pour tout entier $k \geqslant 1$, le coefficient situé à la ligne i et à la colonne j de la matrice M^k est égal au **nombre de chemins (ou de chaînes) de longueur k reliant le sommet i au sommet j**.

Démonstration 6 : L'idée de la démonstration repose sur la définition du produit matriciel. Regardons le cas $k = 2$: Le coefficient $(M^2)_{i,j}$ s'obtient en multipliant la ligne i par la colonne j de M :

$$(M^2)_{i,j} = \sum_{p=1}^{n} m_{i,p} \times m_{p,j}$$

Le produit $m_{i,p} \times m_{p,j}$ vaut 1 si et seulement si $m_{i,p} = 1$ ET $m_{p,j} = 1$ (c'est-à-dire s'il existe une arête de i vers p et une arête de p vers j). Faire la somme sur tous les sommets intermédiaires p possibles revient donc exactement à compter tous les chemins de longueur 2 allant de i à j. □

▪ **Exemple 23 :** Considérons le graphe orienté suivant :



Notons M sa matrice d'adjacence. On a

$$M = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \quad M^2 = \begin{pmatrix} 0 & 0 & 0 & \mathbf{2} \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \quad M^3 = \begin{pmatrix} \mathbf{2} & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & \mathbf{2} \end{pmatrix}$$

Interprétations :

- Le coefficient situé ligne 1, colonne 4 de M^2 vaut **2**. Le théorème affirme qu'il existe exactement 2 chemins de longueur 2 pour aller de 1 vers 4. On vérifie sur le graphe que ce sont les chemins $(1, 2, 4)$ et $(1, 3, 4)$.
- Les coefficients sur la diagonale de M^3 permettent de dénombrer les **circuits** de longueur 3.
  Par exemple, le coefficient $(M^3)_{1,1} = \mathbf{2}$ indique qu'il existe 2 chemins de longueur 3 partant et revenant au sommet 1 : ce sont les circuits $(1, 2, 4, 1)$ et $(1, 3, 4, 1)$. De même, $(M^3)_{4,4} = \mathbf{2}$ correspond aux circuits $(4, 1, 2, 4)$ et $(4, 1, 3, 4)$.

3.4 Lien avec la connexité

On peut utiliser les puissances successives de la matrice pour savoir de manière purement algébrique si un graphe est (fortement) connexe.

Propriété 9 — Caractérisation matricielle de la connexité : Soit $\mathcal{G}$ un graphe (orienté ou non) d'ordre n et M sa matrice d'adjacence.

Soit T la matrice définie par

$$T = M + M^2 + M^3 + \cdots + M^{n-1} = \sum_{k=1}^{n-1} M^k$$

Le graphe $\mathcal{G}$ est (fortement) connexe si et seulement si la matrice T **ne contient aucun coefficient nul**. (Chaque coefficient $t_{i,j}$ est strictement positif).

Démonstration 7 : **Idée :** En effet, le produit de deux matrices à coefficients positifs est positifs.

Par ailleurs, s'il existe un chemin reliant deux sommets i et j, alors il existe un chemin de longueur $n-1$ (il passerait par tous les sommets du graphe). Le coefficient (i, j) de l'une des matrices M^k serait alors non nul.

Si le coefficient $t_{i,j}$ de la matrice T est égal à 0, cela signifie qu'il n'existe strictement aucun chemin (de quelque longueur que ce soit) pour aller de i à j. Le graphe n'est donc pas connexe. □

4 Algorithmique et Programmation

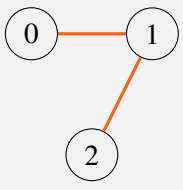
Pour qu'un ordinateur puisse manipuler un graphe, il faut traduire notre dessin géométrique en une structure de données informatique.

4.1 Implémentation d'un graphe en Python

La matrice d'adjacence (Tableaux Numpy)

C'est la traduction directe de la matrice vue dans la section précédente. On utilise généralement la bibliothèque numpy pour créer un tableau bidimensionnel.

▪ **Exemple 24 :** On considère le graphe suivant et son implémentation sous forme de matrice d'adjacence.



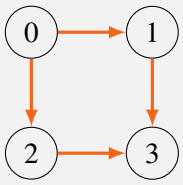

```python
import numpy as np

# Matrice M du graphe :
M = np.array([
    [0, 1, 0],  # Ligne du sommet 0
    [1, 0, 1],  # Ligne du sommet 1
    [0, 1, 0]   # Ligne du sommet 2
])
```

La liste d'adjacence (Liste de listes)

Pour de très grands graphes avec peu d'arêtes (comme un réseau routier), la matrice d'adjacence contient énormément de zéros qui encombrent la mémoire inutilement. On préfère alors utiliser une **liste de listes**. La i-ème sous-liste contient uniquement les sommets voisins (ou successeurs) du sommet i.

▪ **Exemple 25 :** On considère le graphe suivant et son implémentation sous forme de liste d'adjacence.




```python
# Liste d'adjacence du graphe oriente
L = [
    [1, 2],  # Successeurs du sommet 0
    [3],     # Successeurs du sommet 1
    [3],     # Successeurs du sommet 2
    []       # Successeurs du sommet 3 (aucun)
]
# Pour acceder aux voisins du sommet 0 : L[0]
```

Note culturelle : Si les sommets ne sont pas numérotés par des entiers (par exemple s'ils s'appellent "Paris", "Lyon"), on utilise en informatique une structure appelée **dictionnaire d'adjacence** pour lier chaque nom à ses voisins.

4.2 Algorithme de Dijkstra

Principe

L'algorithme de Dijkstra permet de trouver le **plus court chemin** entre un sommet de départ et tous les autres sommets d'un graphe pondéré, à condition que **tous les poids soient positifs**.

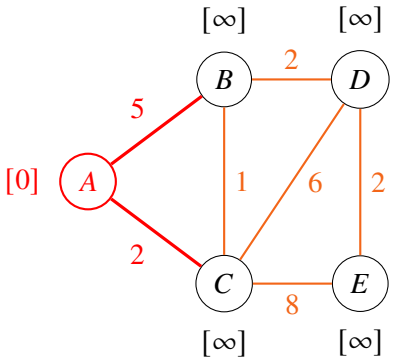
C'est un algorithme glouton : à chaque étape, on sélectionne le sommet non traité dont la distance provisoire est la plus petite, on le valide définitivement, et on met à jour les distances de ses voisins non encore validés.

Exemple d'exécution pas-à-pas

Considérons le graphe **non orienté** pondéré ci-dessous. Nous cherchons le plus court chemin partant du sommet **A**.

À chaque étape, nous noterons la distance minimale connue pour atteindre un sommet entre crochets. Si un sommet est inatteignable ou non encore exploré, sa distance est notée ∞.

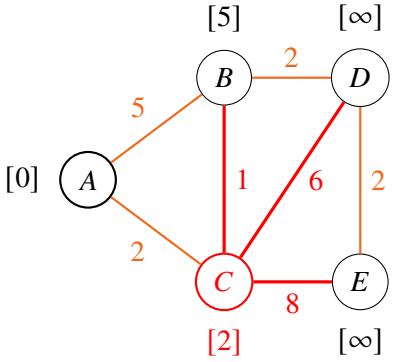
Initialisation : On affecte le poids 0 au sommet de départ A, et ∞ à tous les autres. On place A dans la liste des sommets visités (en rouge) et on regarde ses voisins directs.



Sélection	A	B	C	D	E
Init.	**0**	∞	∞	∞	∞
A (0)	X	5 (A)	2 (A)	∞	∞

*Explication : Les voisins de A sont B et C. Passer par A coûte 0 + 5 = 5 pour aller en B, et 0 + 2 = 2 pour aller en C. Parmi les sommets non fixés (B, C, D, E), c'est **C** qui a la plus petite distance (2). On le sélectionne pour l'étape suivante.*

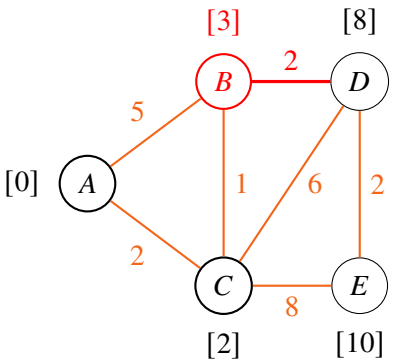
Étape 1 : On sélectionne C. La distance pour aller en C est fixée à 2. On explore ses voisins non fixés (B, D, E). Le sommet A est ignoré car déjà validé.



Sélection	A	B	C	D	E
Init.	**0**	∞	∞	∞	∞
A (0)	X	5 (A)	**2 (A)**	∞	∞
C (2)	X	**3 (C)**	X	8 (C)	10 (C)

*Explication : Pour B, l'ancien chemin (direct par A) coûtait 5. Le nouveau chemin (en passant par C) coûte 2 + 1 = 3. C'est plus court ! On **met à jour** B. On calcule aussi pour D (2 + 6 = 8) et E (2 + 8 = 10). Le plus petit sommet non fixé est **B** (3).*

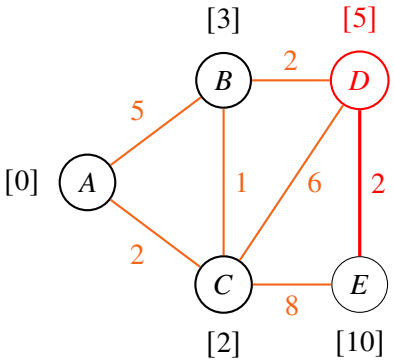
Étape 2 : On sélectionne B (distance 3). Son seul voisin non exploré est D.



Sélection	A	B	C	D	E
Init.	**0**	∞	∞	∞	∞
A (0)	X	5 (A)	**2 (A)**	∞	∞
C (2)	X	**3 (C)**	X	8 (C)	10 (C)
B (3)	X	X	X	**5 (B)**	10 (C)

*Explication : En passant par B, atteindre D coûte 3+2 = 5. C'est meilleur que l'ancien chemin depuis C qui coûtait 8 ! On met à jour la colonne de D. Le prochain sommet le plus proche est **D** (5).*

Étape 3 : On sélectionne D (distance 5). Son seul voisin non fixé est E.



Sélection	A	B	C	D	E
Init.	**0**	∞	∞	∞	∞
A (0)	X	5 (A)	**2 (A)**	∞	∞
C (2)	X	**3 (C)**	X	8 (C)	10 (C)
B (3)	X	X	X	**5 (B)**	10 (C)
D (5)	X	X	X	X	**7 (D)**

Explication : En passant par D, le sommet E coûte 5 + 2 = 7. C'est mieux que l'ancien 10. On met à jour E, qui est le dernier sommet. Le tableau est entièrement rempli !

Conclusion et lecture du résultat :
Pour trouver le plus court chemin vers E, on part de sa dernière valeur dans le tableau et on "remonte" de parenthèse en parenthèse :

- Pour aller en E, il faut venir de **D** (coût 7).
- Pour aller en D, il faut venir de **B** (coût 5).
- Pour aller en B, il faut venir de **C** (coût 3).
- Pour aller en C, il faut venir de **A** (coût 2).

Le plus court chemin de A vers E est donc : **A** → **C** → **B** → **D** → **E**, et sa longueur (son poids total) est de **7**.

R L'algorithme de Dijkstra, dans sa version complète, calcule les plus courts chemins vers **tous** les sommets du graphe.

Cependant, dans la majorité des cas concrets (comme un GPS), on cherche uniquement le chemin vers **un sommet de destination précis**. Dans ce cas, **l'algorithme s'arrête dès que le sommet de destination est sélectionné** (validé définitivement dans la colonne de gauche). En effet, puisque tous les poids du graphe sont positifs, aucun chemin ultérieur ne pourra faire baisser cette distance.

Lien avec notre exemple : Si notre objectif final n'avait pas été le sommet E, mais uniquement le sommet **D**, nous aurions pu stopper l'algorithme au tout début de l'**Étape 3**. Au moment où le sommet **D (5)** est sélectionné comme étant le plus proche, sa distance est définitivement validée. Il est alors totalement inutile de calculer les distances de ses voisins (ici E) ou de finir le tableau !

Implémentation en Python

Pour programmer cet algorithme sur notre graphe pondéré, nous allons utiliser une liste d'adjacence. Puisque les arêtes ont un poids, chaque voisin sera représenté par un **couple** `(sommet_voisin, poids)`.

Comme Python indexe ses listes à partir de 0, nous utiliserons la correspondance suivante pour nos sommets : $A = 0, B = 1, C = 2, D = 3, E = 4$.

```python
def dijkstra(graphe, depart):
    n = len(graphe)
    # Initialisation des tableaux (infinis partout sauf au depart)
    distances = [float('inf')] * n
    distances[depart] = 0

    # Pour retenir le chemin (le tableau des parentheses)
    predecesseurs = [None] * n

    # Pour savoir quels sommets ont deja ete valides
    visites = [False] * n

    # On doit valider les n sommets
    for _ in range(n):
        # 1. Trouver le sommet NON VISITE le plus proche
        dist_min = float('inf')
        u = -1
        for i in range(n):
            if not visites[i] and distances[i] < dist_min:
                dist_min = distances[i]
                u = i

        # Si aucun sommet atteignable n'est trouve, on arrete
        if u == -1:
            break

        # 2. On valide ce sommet definitivement
        visites[u] = True
```

```
        # 3. Mise a jour des distances de ses voisins
        for voisin, poids in graphe[u]:
            if not visites[voisin]:
                nouvelle_dist = distances[u] + poids
                # Si le nouveau chemin est meilleur, on met a jour !
                if nouvelle_dist < distances[voisin]:
                    distances[voisin] = nouvelle_dist
                    predecesseurs[voisin] = u

    return distances, predecesseurs

# --- TEST SUR LE GRAPHE DE L'EXEMPLE ---
# 0:A, 1:B, 2:C, 3:D, 4:E
graphe_exemple = [
    [(1, 5), (2, 2)],                # Voisins de A (0)
    [(0, 5), (2, 1), (3, 2)],        # Voisins de B (1)
    [(0, 2), (1, 1), (3, 6), (4, 8)], # Voisins de C (2)
    [(1, 2), (2, 6), (4, 2)],        # Voisins de D (3)
    [(2, 8), (3, 2)]                 # Voisins de E (4)
]

# Execution en partant de A (sommet 0)
dist, pred = dijkstra(graphe_exemple, 0)

print("Distances minimales depuis A :", dist)
print("Tableau des predecesseurs    :", pred)
```

Résultat de l'exécution

Lorsque l'on exécute ce code, la console Python affiche exactement le résultat de la dernière ligne de notre tableau manuel :

```
Distances minimales depuis A : [0, 3, 2, 5, 7]
Tableau des predecesseurs    : [None, 2, 0, 1, 3]
```

Lecture de la sortie informatique :

- La liste des distances nous indique que pour aller au sommet 4 (soit E), la distance minimale est de **7**.
- Pour reconstruire le chemin vers 4, on regarde les prédécesseurs à l'envers :
  - `pred[4]` est **3** (E vient de D)
  - `pred[3]` est **1** (D vient de B)
  - `pred[1]` est **2** (B vient de C)
  - `pred[2]` est **0** (C vient de A)

  On retrouve bien notre chemin optimal : $0 \to 2 \to 1 \to 3 \to 4$ (soit $A \to C \to B \to D \to E$).