Chapitre 18 - Les graphes – Partie 1

De nombreuses situations peuvent être modélisées par des graphes. On peut citer les exemples suivants :

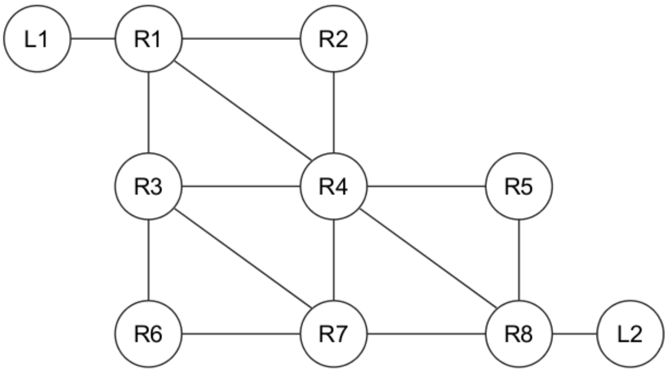


Figure 1 : Graphe pour modéliser un réseau de routeurs

Figure 2 : Graphe pour modéliser un réseau social

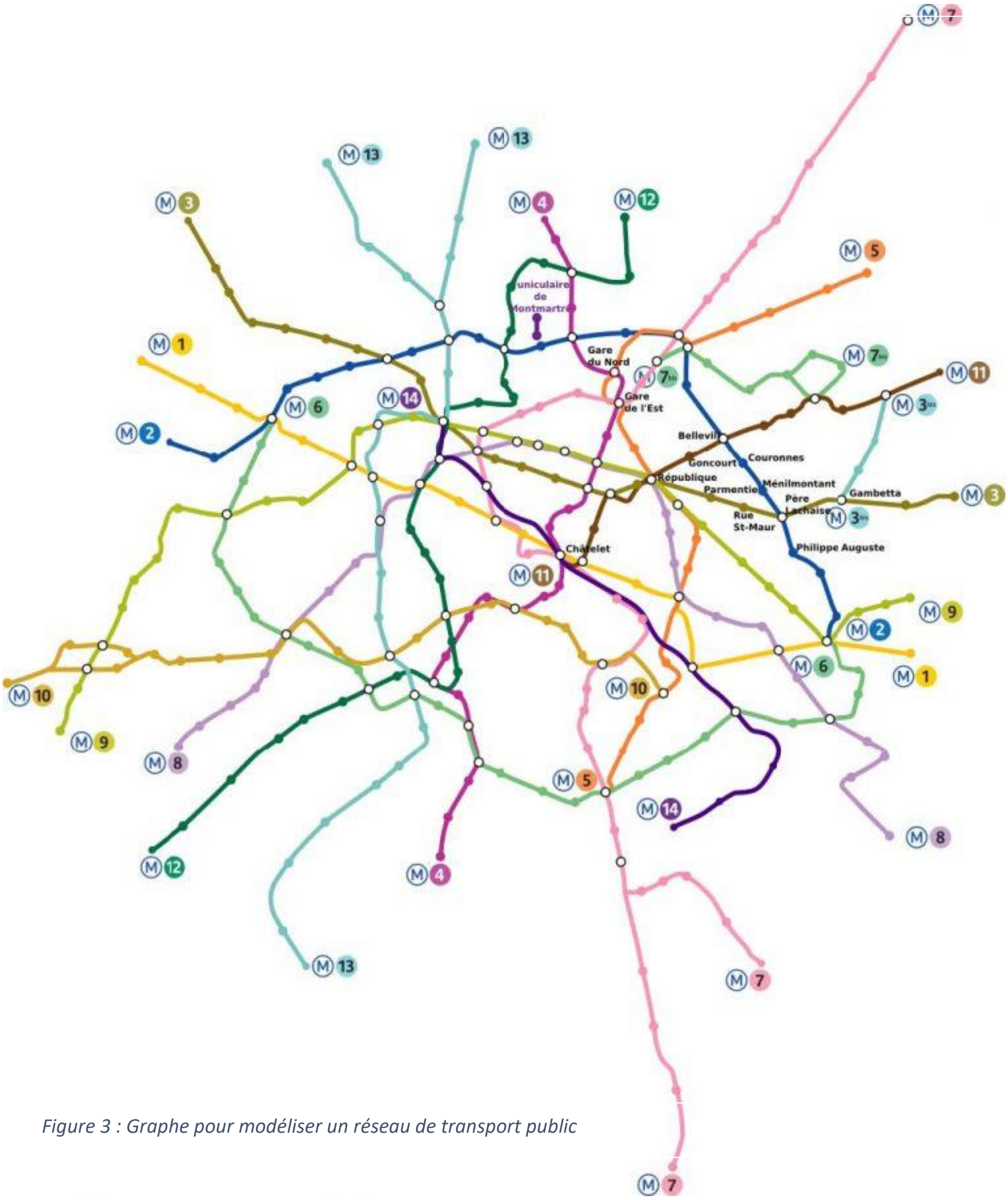


Figure 3 : Graphe pour modéliser un réseau de transport public

L'étude de la théorie des graphes est un champ très vaste des mathématiques. Ici on l'aborde du point de vue informatique. Dans ce chapitre, on se posera les questions suivantes :

⇨ Comment implémente-t-on un graphe ? Utilise-t-on des listes, les dictionnaires, la POO ?

⇨ L'implémentation étant définie, comment s'y prend-on

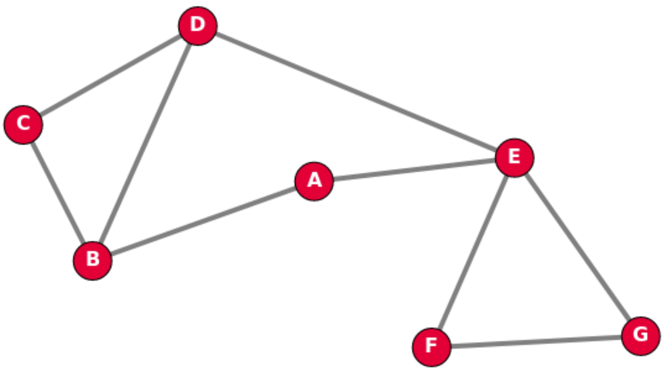
- pour parcourir tous les nœuds du graphe ?
- pour rechercher les différents chemins qui mènent à un nœud donné ?
- pour rechercher le chemin le plus court.

1- Vocabulaire :

Point Cours :

En général, un graphe est un ensemble d'objets, appelés ou parfois (*vertex* or *nodes* en anglais) reliés par des ou *arcs* ((*edges* en anglais)). Ce graphe peut être **non-orienté** ou **orienté**

a. Graphe non orientee :



Dans un graphe **non-orienté**, les *arêtes* peuvent être empruntées dans les deux sens.

Une chaîne est une suite de sommets reliés par des arêtes, comme C - B - A - E par exemple. La longueur de cette chaîne est alors 3, soit le nombre d'arêtes empruntées.

Q1. : Quelle est la longueur de la chaine F-E-D ? :

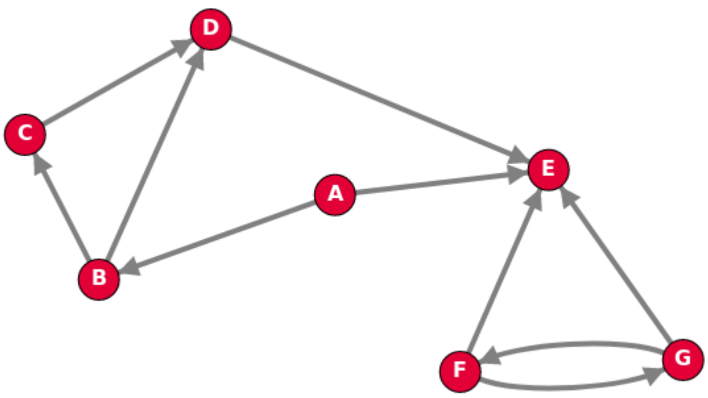
Les sommets B et E sont *adjacents* au sommet A, ce sont les *voisins* de A.

Q2. : Quels sont les voisins de D ? :

Q3. : Compléter : « Les nœuds G, F et A sont adjacents au nœud …. »

Q4. : Le graphe des relations d'un individu sur *Facebook* est-il non-orienté ? Si on est « ami » avec quelqu'un la réciproque est-elle vraie ?

b. Graphe orientee :



Dans un graphe **orienté**, les *arcs* ne peuvent être empruntés que dans le sens de la flèche, et un *chemin* est une suite de sommets reliés par des arcs, comme B → C → D → E par exemple.

Les sommets C et D sont *adjacents* au sommet B (mais pas A !), ce sont les *voisins* de B.

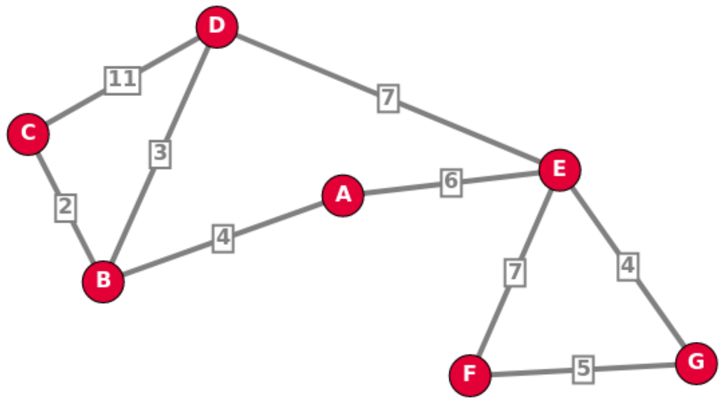
Q5. : Peut-on aller de C à G ? :

Q6. : Quels sont les voisins de D ? :

Q7. : Compléter : « Le nœud E est adjacents aux nœuds , il est le voisin des nœuds »

Q8. : Le graphe des relations d'un individu sur *X* est-il orienté ? Peut-on «suivre» quelqu'un sans que cela soit réciproque ?

c. GRAPHE PONDERE :

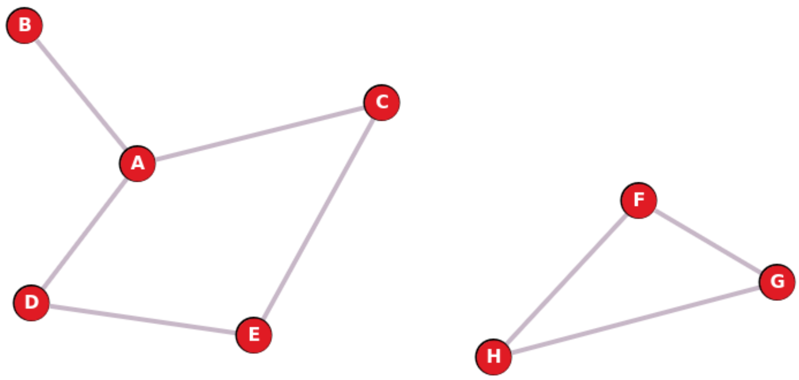


Un graphe est **pondéré** (ou valué) si on attribue à chaque arête une valeur numérique (la plupart du temps positive), qu'on appelle *mesure*, *poids*, *coût* ou *valuation*.

Par exemple:

- dans le protocole OSPF, on pondère les liaisons entre routeurs par le coût;
- dans un réseau routier entre plusieurs villes, on pondère par les distances.

d. CONNEXITE D'UN GRAPHE :



Un graphe est **connexe** s'il est d'un seul tenant. C'est-à-dire si n'importe quelle paire de sommets peut toujours être reliée par une chaîne. Autrement dit, un graphe est connexe s'il est «en un seul morceau».

Par exemple, le graphe ci-contre n'est pas connexe. Il n'existe pas de chaîne entre les sommets A et F par exemple.

Pour modéliser un graphe, il faut établir par convention une manière de donner les renseignements suivants :

- qui sont les sommets ?
- pour chaque sommet, quels sont ses voisins ? (et éventuellement quel poids porte l'arête qui les relie)

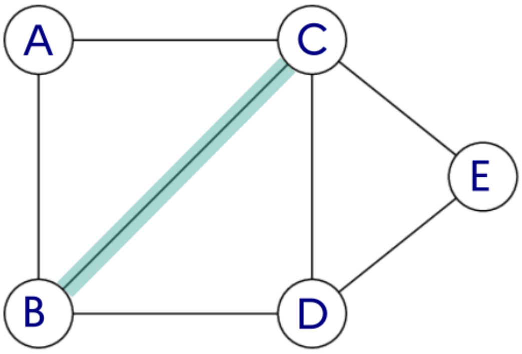
1- IMPLEMENTATION PAR MATRICES D'ADJACENCE :

Principe :

- On classe les sommets (en les numérotant, ou par ordre alphabétique)
- On représente les arêtes dans une matrice, c'est-à-dire un tableau à deux dimensions où on inscrit un en ligne i et colonne j si les sommets de rang i et de rang j sont **voisins** (dits aussi *adjacents*).

Ce tableau s'appelle une **matrice d'adjacence** (on aurait très bien pu l'appeler aussi *matrice de voisinage*)

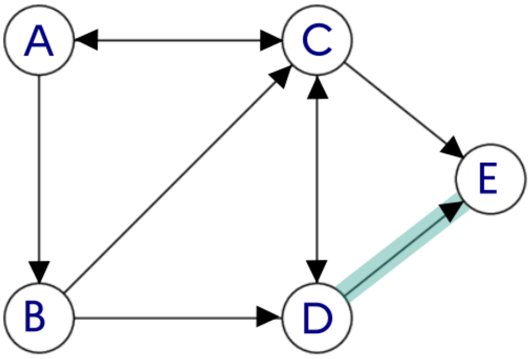
Exemple d’un graphe non orienté :



$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Dans ce graphe non orienté, comme B est voisin de C, C est aussi voisin de B, ce qui signifie que l'arête qui relie B et C va donner lieu à deux "1" dans la matrice, situé de part et d'autre de la diagonale descendante (un mathématicien parlera de matrice *symétrique*)

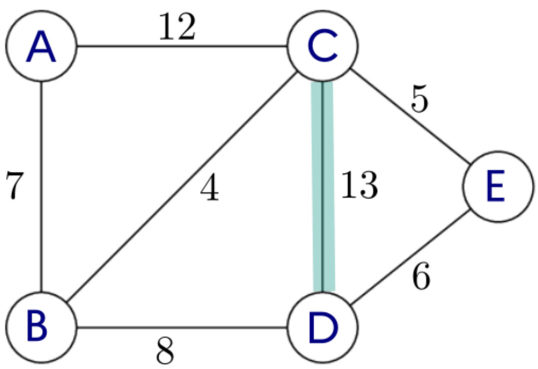
Exemple d’un graphe orienté :



$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Comme le graphe est orienté, la matrice n'est pas forcément symétrique (il faudrait que tous les liens soient réciproques pour qu'elle le soit).

Exemple d’un graphe pondéré et non orienté :



$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} 0 & 7 & 12 & 0 & 0 \\ 7 & 0 & 4 & 8 & 0 \\ 12 & 4 & 0 & 13 & 5 \\ 0 & 8 & 13 & 0 & 6 \\ 0 & 0 & 5 & 6 & 0 \end{pmatrix} \end{array}$$

Implémentation python des matrices d’adjacences : Une matrice se représente naturellement par une liste de listes.

$$\begin{pmatrix} 0 & 7 & 12 & 0 & 0 \\ 7 & 0 & 4 & 8 & 0 \\ 12 & 4 & 0 & 13 & 5 \\ 0 & 8 & 13 & 0 & 6 \\ 0 & 0 & 5 & 6 & 0 \end{pmatrix}$$

Implémenté par la liste de listes :

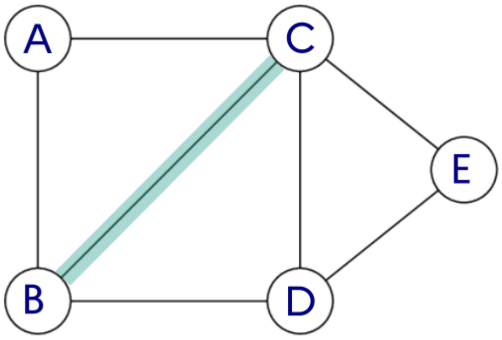
```
g = [ [ 0, 7,12, 0, 0],
      [ 7, 0, 4, 8, 0],
      [12, 4, 0,13, 5],
      [ 0, 8,13, 0, 6],
      [ 0, 0, 5, 6, 0]
]
```

2- IMPLEMENTATION PAR LISTES D'ADJACENCE :

Principe :

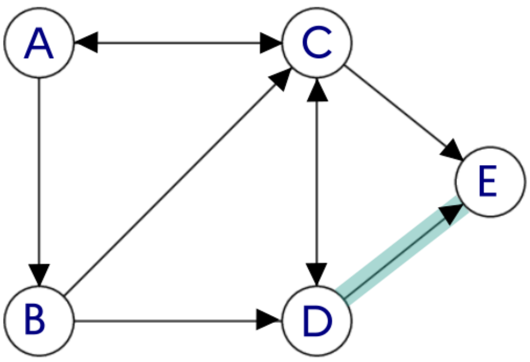
- On associe à chaque sommet sa liste des voisins (c'est-à-dire les sommets adjacents). On utilise pour cela un dictionnaire dont les clés sont les sommets et les valeurs les listes des voisins.
- Dans le cas d'un graphe orienté on associe à chaque sommet la liste des *successeurs* (ou bien des *prédécesseurs*, au choix).

Exemple d'un graphe non orienté :



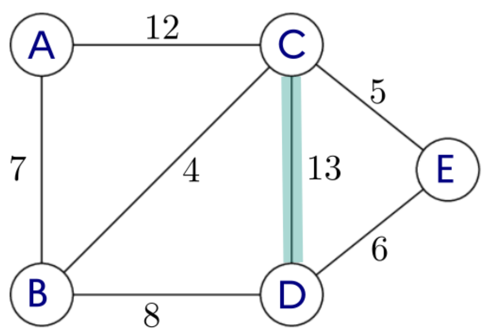

```
g = {'A': ['B', 'C'],
     'B': ['A', 'C' , 'D'],
     'C': ['A', 'B', 'D' , 'E'],
     'D': ['B', 'C', 'E'],
     'E': ['C', 'D']
}
```

Exemple d'un graphe orienté :




```
g = {'A': ['B', 'C'],
     'B': ['C' , 'D'],
     'C': ['A', 'D' , 'E'],
     'D': ['C' , 'E'],
     'E': []
}
```

Exemple d'un graphe pondéré et non orienté :




```
g = {'A': [('B',7) , ('C',12)],
     'B': [('A',7), ('C', 4) , ('D', 8)],
     'C': [('A',12) , ('B',4) , ('D', 13) , ('E', 5)],
     'D': [('B',8) , ('C',13) , ('E', 6)],
     'E': [('C', 5) , ('D', 6)]
}
```