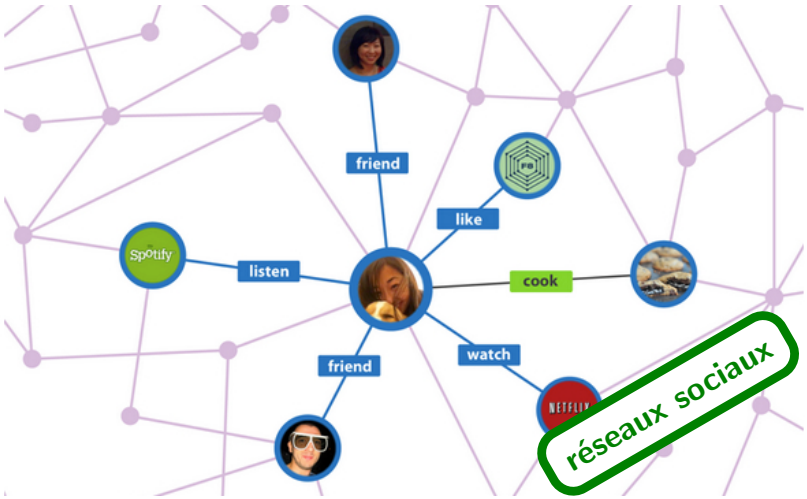
Algorithmique des graphes

1 — Les bases

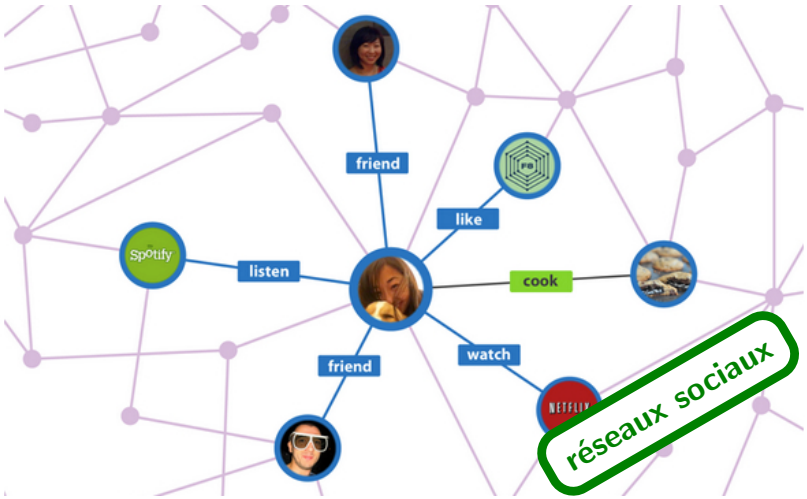
Marie-Pierre Béal (Cours d'Anthony Labarre)

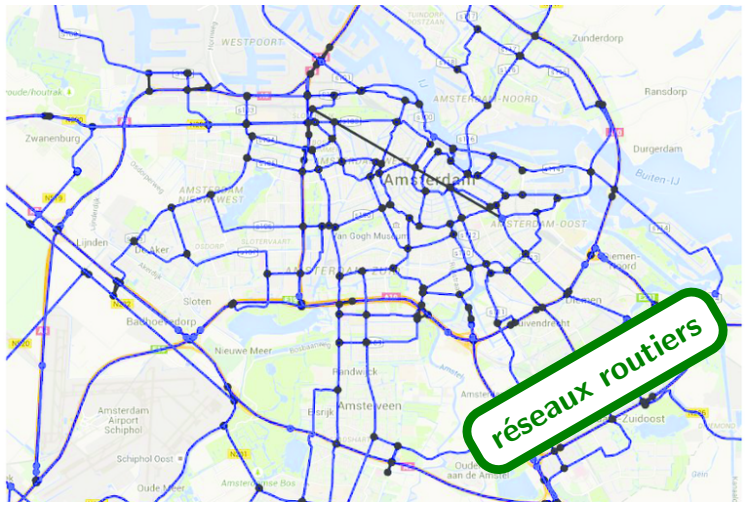
Université
Gustave Eiffel

Graphes : exemples informels

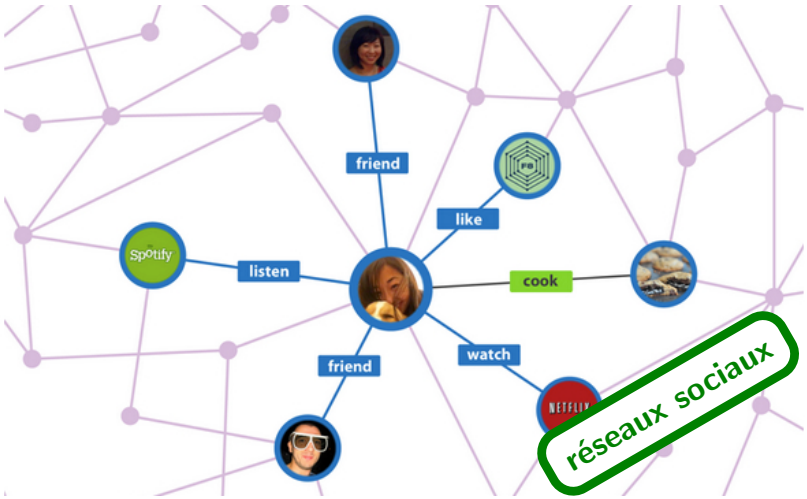


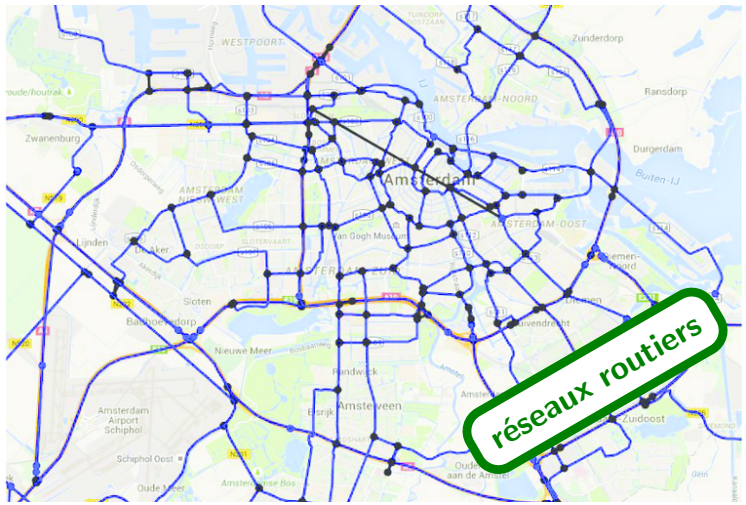
Graphes : exemples informels

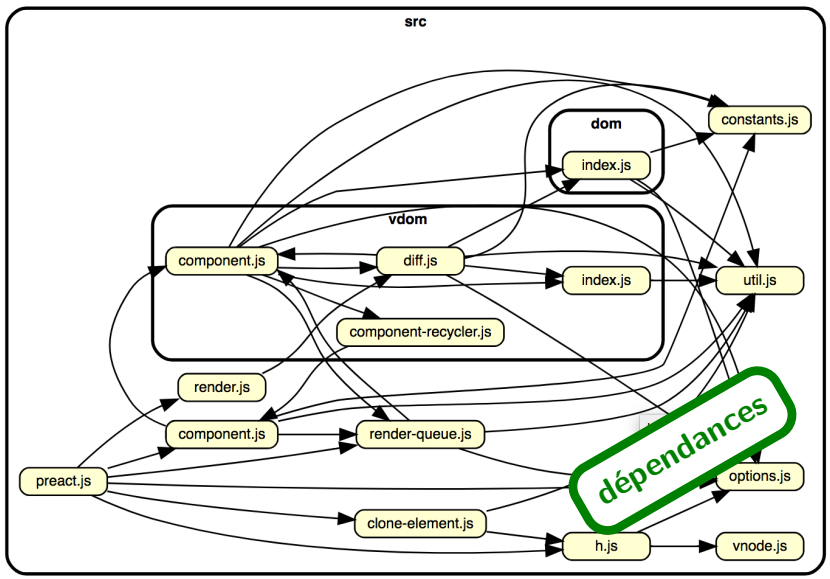




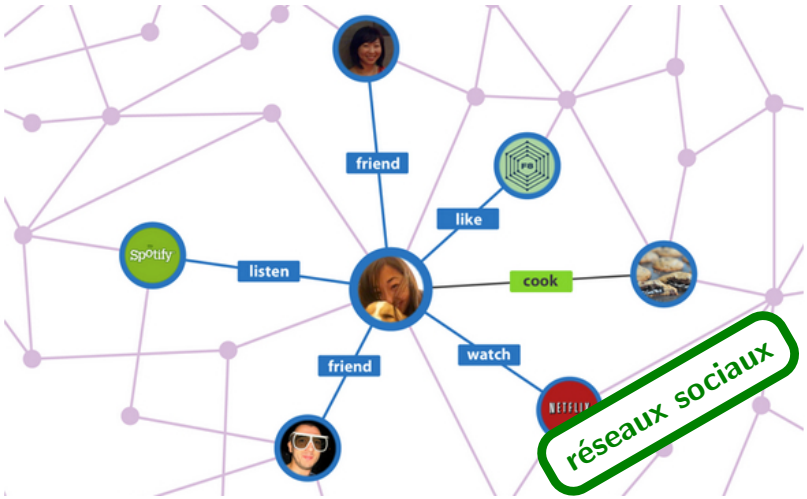
Graphes : exemples informels

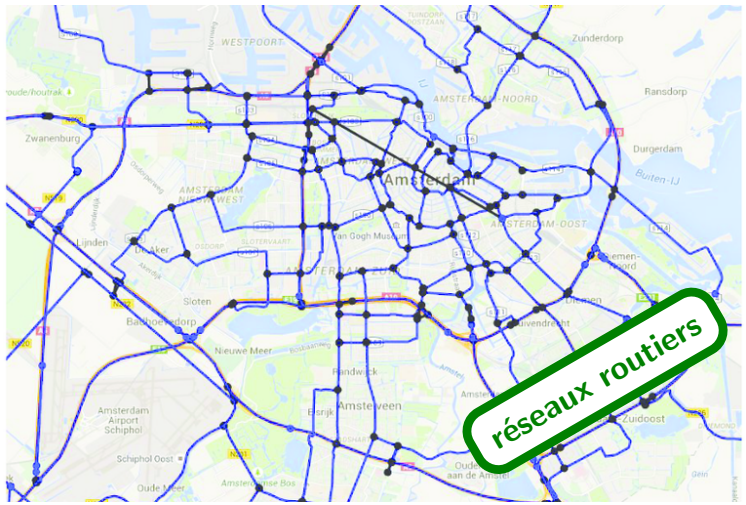


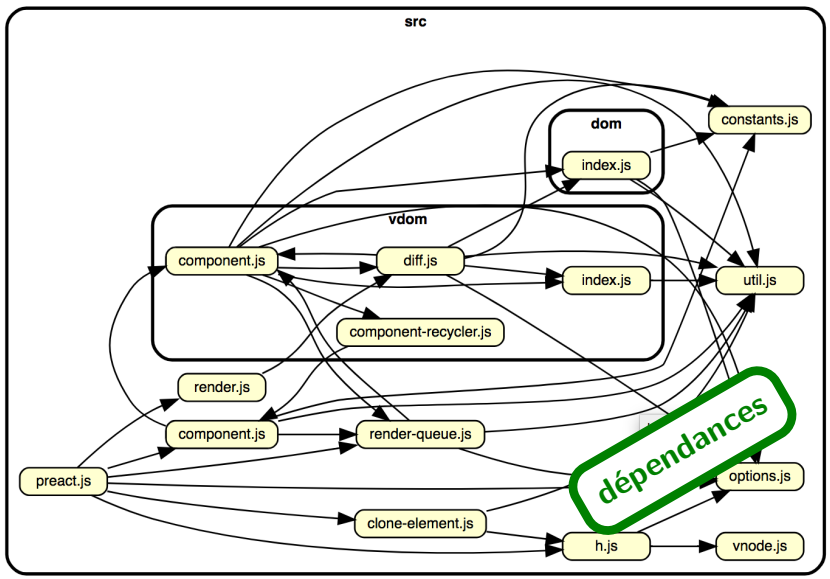


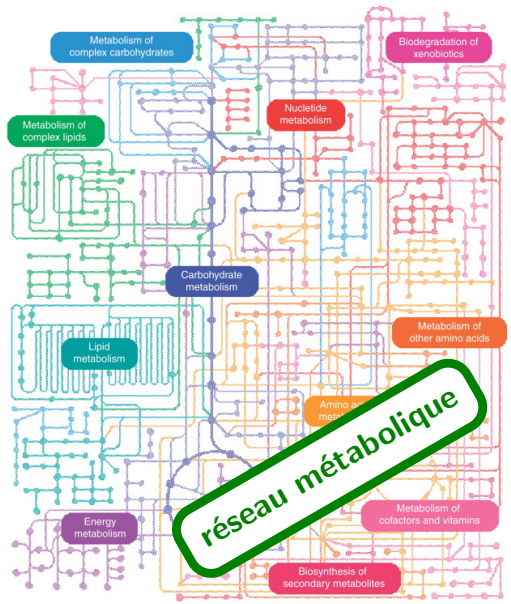


Graphes : exemples informels

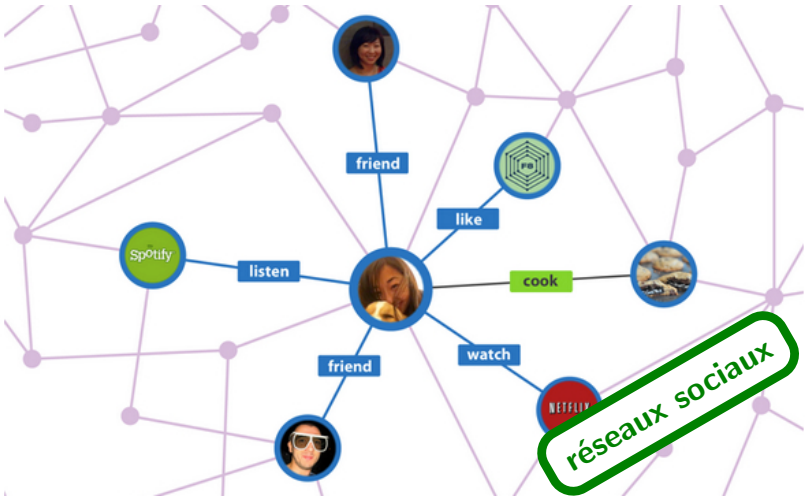


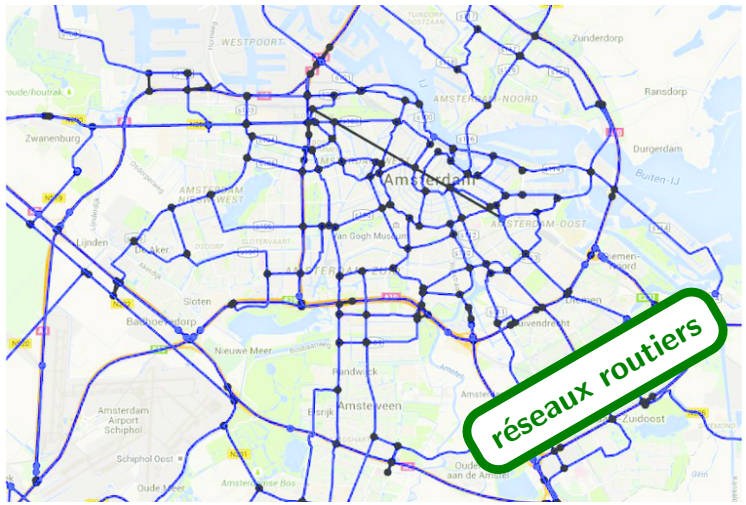


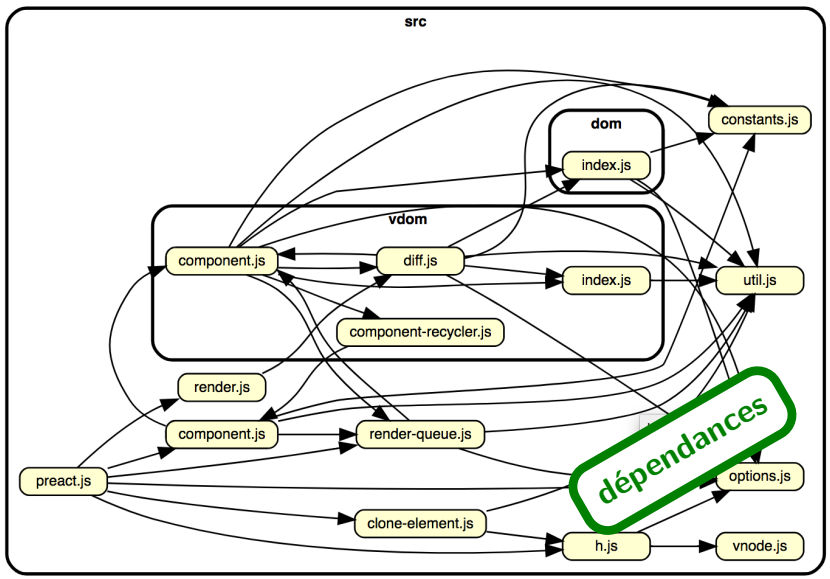


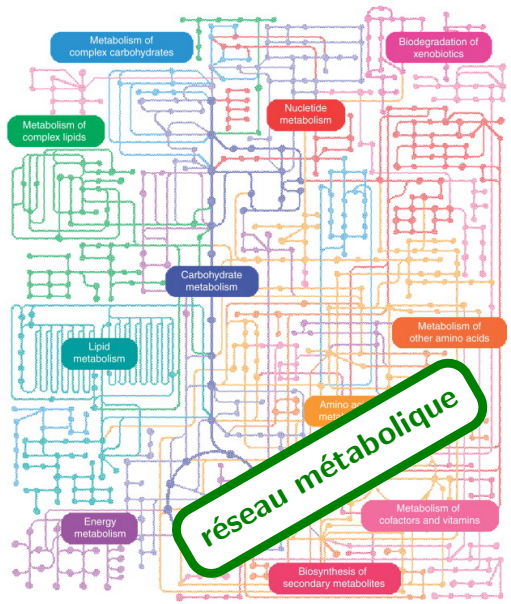


Graphes : exemples informels









$\Rightarrow$ Graphe = éléments reliés par une certaine relation

Concepts de base : graphe non-orienté

Définition 1

Un **graphe (non-orienté)** est un couple $G = (V, E)$, où :

- V est un ensemble de **sommets** ;
- E un ensemble de paires d'éléments de V appelées **arêtes**.

On utilisera aussi les notations $V(G)$ et $E(G)$ pour bien distinguer le graphe G d'un autre graphe.

Concepts de base : graphe non-orienté

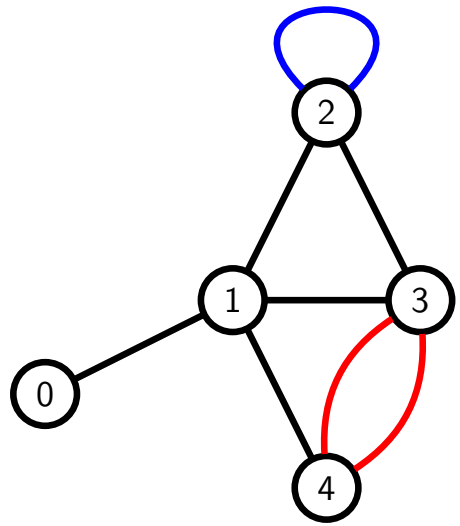
Définition 1

Un **graphe (non-orienté)** est un couple $G = (V, E)$, où :

- V est un ensemble de **sommets** ;
- E un ensemble de paires d'éléments de V appelées **arêtes**.

On utilisera aussi les notations $V(G)$ et $E(G)$ pour bien distinguer le graphe G d'un autre graphe.

Exemple 1



Concepts de base : graphe non-orienté

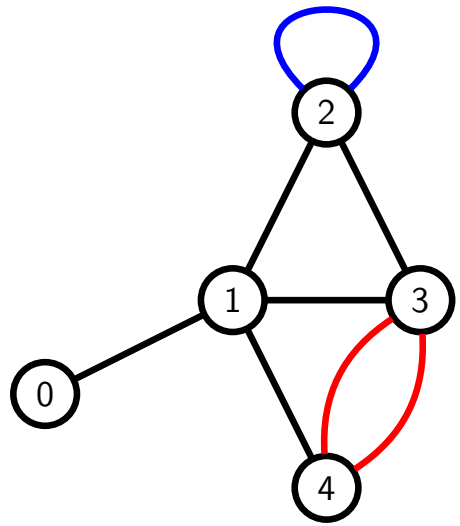
Définition 1

Un **graphe (non-orienté)** est un couple $G = (V, E)$, où :

- V est un ensemble de **sommets** ;
- E un ensemble de paires d'éléments de V appelées **arêtes**.

On utilisera aussi les notations $V(G)$ et $E(G)$ pour bien distinguer le graphe G d'un autre graphe.
G est **simple** s'il ne contient ni **arêtes parallèles** ni **boucles**.

Exemple 1



Concepts de base : graphe non-orienté

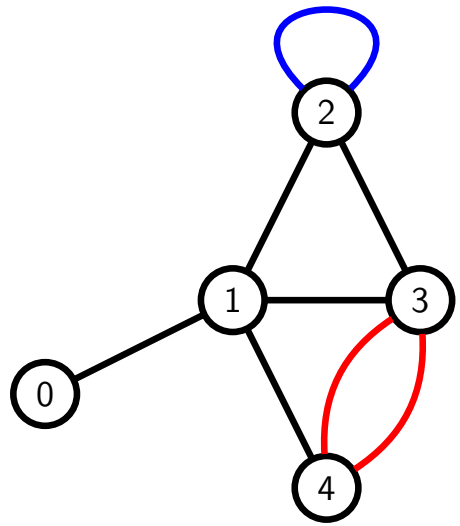
Définition 1

Un **graphe (non-orienté)** est un couple $G = (V, E)$, où :

- V est un ensemble de **sommets** ;
- E un ensemble de paires d'éléments de V appelées **arêtes**.

On utilisera aussi les notations $V(G)$ et $E(G)$ pour bien distinguer le graphe G d'un autre graphe.
G est **simple** s'il ne contient ni **arêtes parallèles** ni **boucles**.

Exemple 1



- Une arête $\{u, v\}$ relie deux sommets **adjacents** ou **voisins** ; elle est **incidente** à u et v, qui sont ses **extrémités** ;

Concepts de base : graphe non-orienté

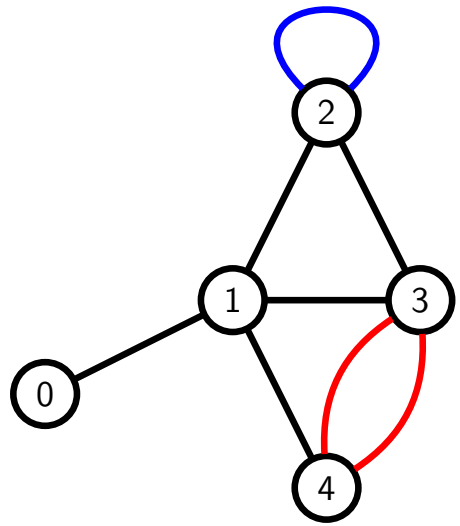
Définition 1

Un **graphe (non-orienté)** est un couple $G = (V, E)$, où :

- V est un ensemble de **sommets** ;
- E un ensemble de paires d'éléments de V appelées **arêtes**.

On utilisera aussi les notations $V(G)$ et $E(G)$ pour bien distinguer le graphe G d'un autre graphe.
G est **simple** s'il ne contient ni **arêtes parallèles** ni **boucles**.

Exemple 1



- Une arête $\{u, v\}$ relie deux sommets **adjacents** ou **voisins** ; elle est **incidente** à u et v, qui sont ses **extrémités** ;
- Le **voisinage** d'un sommet v est l'ensemble de ses voisins :

$$N_G(v) = \{u \mid \{u, v\} \in E(G)\}$$

Concepts de base : graphe non-orienté

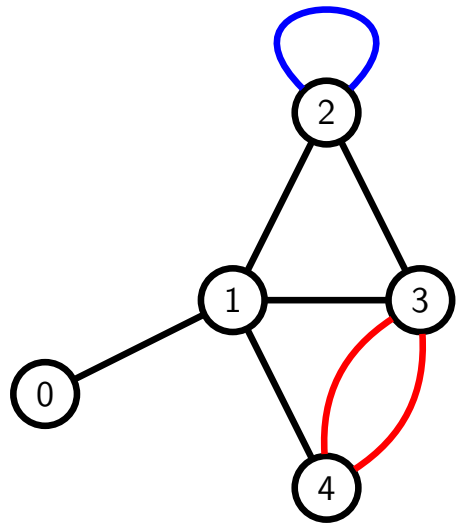
Définition 1

Un **graphe (non-orienté)** est un couple $G = (V, E)$, où :

- V est un ensemble de **sommets** ;
- E un ensemble de paires d'éléments de V appelées **arêtes**.

On utilisera aussi les notations $V(G)$ et $E(G)$ pour bien distinguer le graphe G d'un autre graphe.
G est **simple** s'il ne contient ni **arêtes parallèles** ni **boucles**.

Exemple 1



- Une arête $\{u, v\}$ relie deux sommets **adjacents** ou **voisins** ; elle est **incidente** à u et v, qui sont ses **extrémités** ;
- Le **voisinage** d'un sommet v est l'ensemble de ses voisins :

$$N_G(v) = \{u \mid \{u, v\} \in E(G)\}$$

- Le **degré** du sommet v est la taille de son voisinage, noté $\deg_G(v)$;

Chemins dans un graphe non-orienté

- Un **chemin ou chaîne** dans un graphe **non-orienté** $G = (V, E)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $\{u_i, u_{i+1}\} \in E$ pour $0 \leq i \leq p - 2$. La **longueur du chemin** est p.

Chemins dans un graphe non-orienté

- Un **chemin ou chaîne** dans un graphe **non-orienté** $G = (V, E)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $\{u_i, u_{i+1}\} \in E$ pour $0 \leq i \leq p-2$. La **longueur du chemin** est p.
- Un **chemin élémentaire** dans un graphe **non-orienté** est un chemin dont tous les sommets sont distincts.

Chemins dans un graphe non-orienté

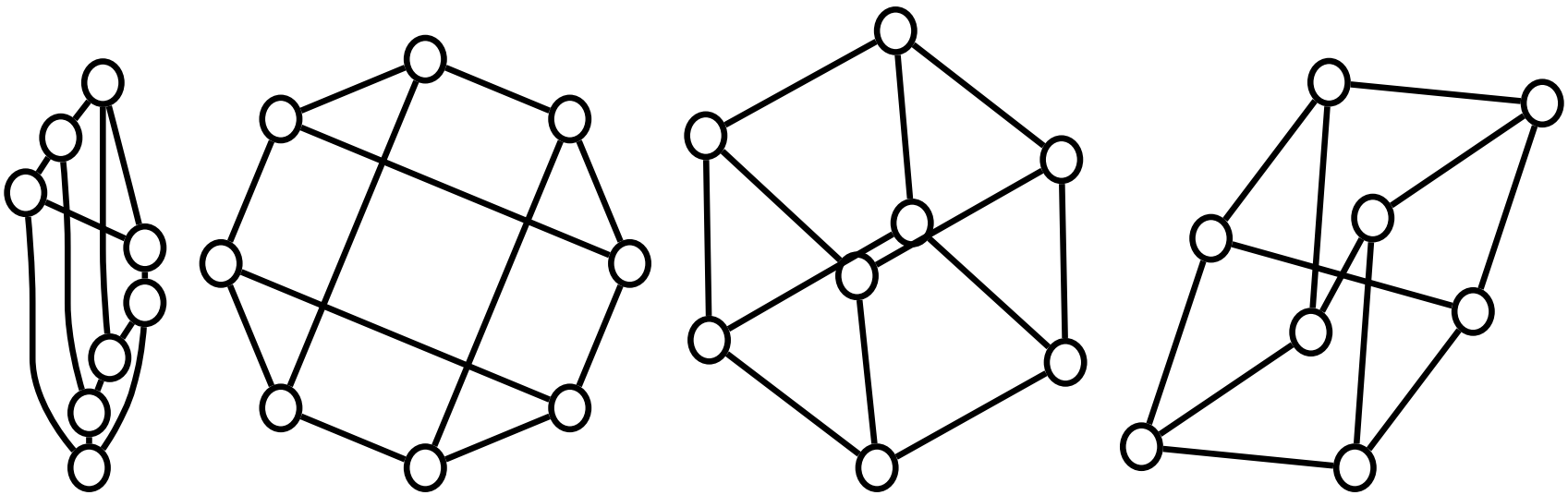
- Un **chemin ou chaîne** dans un graphe **non-orienté** $G = (V, E)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $\{u_i, u_{i+1}\} \in E$ pour $0 \leq i \leq p-2$. La **longueur du chemin** est p.
- Un **chemin élémentaire** dans un graphe **non-orienté** est un chemin dont tous les sommets sont distincts.
- Un **chemin simple** dans un graphe **non-orienté** est un chemin dont toutes les arêtes sont distinctes.

Chemins dans un graphe non-orienté

- Un **chemin ou chaîne** dans un graphe **non-orienté** $G = (V, E)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $\{u_i, u_{i+1}\} \in E$ pour $0 \leq i \leq p-2$. La **longueur du chemin** est p.
- Un **chemin élémentaire** dans un graphe **non-orienté** est un chemin dont tous les sommets sont distincts.
- Un **chemin simple** dans un graphe **non-orienté** est un chemin dont toutes les arêtes sont distinctes.
- Un **cycle** dans un graphe **non-orienté** est un chemin simple dont les deux extrémités sont identiques.

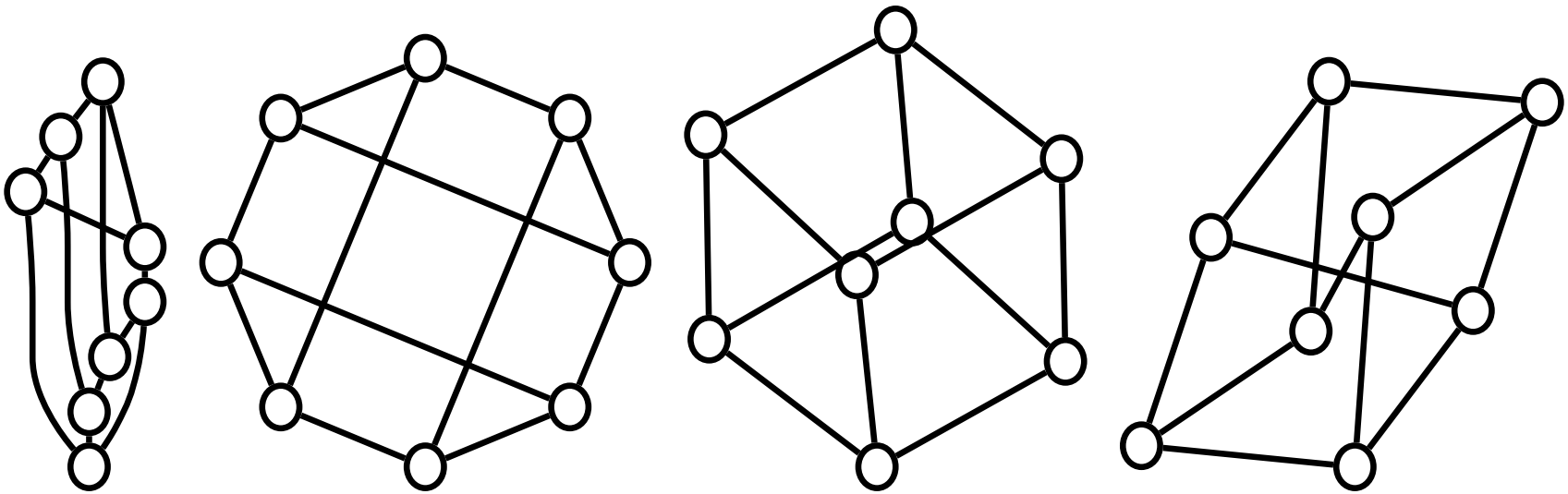
Étiquetages et dessins

Les sommets et les arêtes peuvent être étiquetés, mais ce n'est pas obligatoire. La manière dont on dessine les graphes n'importe pas :



Étiquetages et dessins

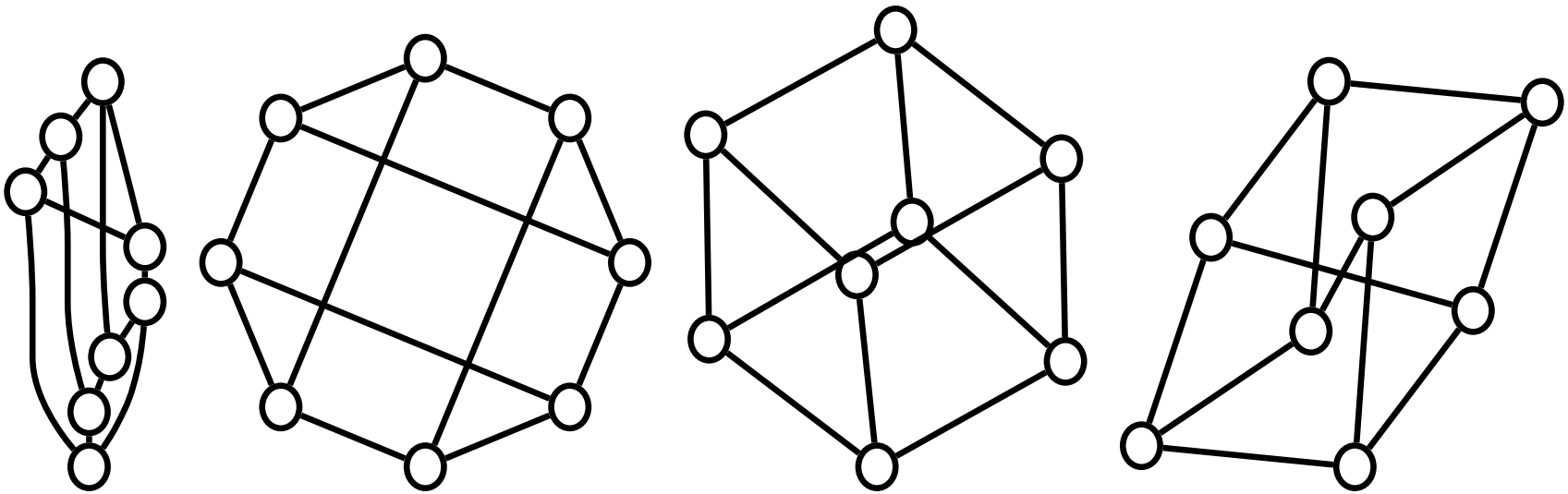
Les sommets et les arêtes peuvent être étiquetés, mais ce n'est pas obligatoire. La manière dont on dessine les graphes n'importe pas :



Les graphes G et H sont **isomorphes** si l'on peut numéroter $V(G)$ et $V(H)$ de telle sorte que $E(G) = E(H)$.

Étiquetages et dessins

Les sommets et les arêtes peuvent être étiquetés, mais ce n'est pas obligatoire. La manière dont on dessine les graphes n'importe pas :

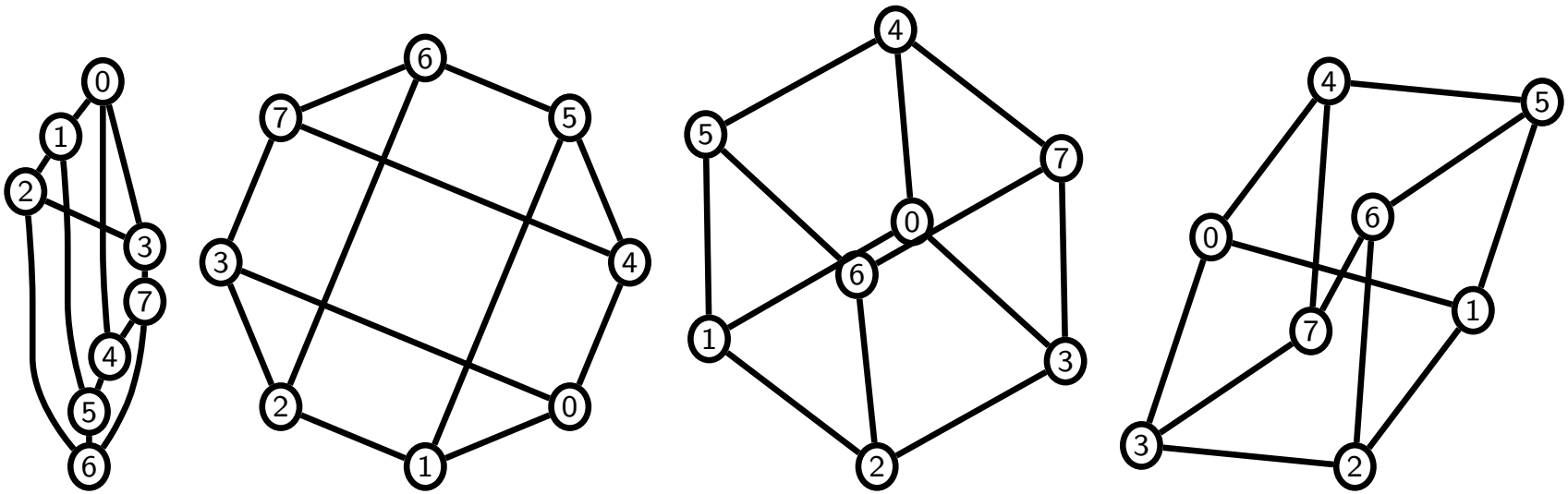


Les graphes G et H sont **isomorphes** si l'on peut numéroter $V(G)$ et $V(H)$ de telle sorte que $E(G) = E(H)$.

On ne connaît pas la complexité du problème de décision associé ; le meilleur algorithme connu est en $\exp((\log n)^{O(1)})$ [1].

Isomorphisme

Voici des étiquetages prouvant que les graphes précédents sont tous isomorphes :



Les arêtes deviennent $\{\{0,1\},\{0,3\},\{0,4\},\ldots,\{6,7\}\}$.

Représentation de graphes en mémoire

- Les deux encodages de graphes les plus fréquents se basent sur des *matrices* ou des *listes* d'adjacence ;

Représentation de graphes en mémoire

- Les deux encodages de graphes les plus fréquents se basent sur des *matrices* ou des *listes* d'adjacence ;
- Le choix de la représentation est important, car :

Représentation de graphes en mémoire

- Les deux encodages de graphes les plus fréquents se basent sur des *matrices* ou des *listes* d'adjacence ;
- Le choix de la représentation est important, car :
  1. la consommation en mémoire n'est pas la même ;

Représentation de graphes en mémoire

- Les deux encodages de graphes les plus fréquents se basent sur des *matrices* ou des *listes* d'adjacence ;
- Le choix de la représentation est important, car :
  1. la consommation en mémoire n'est pas la même ;
  2. la complexité des opérations diffère aussi.

Représentation de graphes en mémoire

- Les deux encodages de graphes les plus fréquents se basent sur des *matrices* ou des *listes* d'adjacence ;
- Le choix de la représentation est important, car :
  1. la consommation en mémoire n'est pas la même ;
  2. la complexité des opérations diffère aussi.
- Le "bon" choix dépend des algorithmes et des applications qui nous intéressent.

Matrice d'adjacence

Définition 2

La **matrice d'adjacence** $A(G)$ du graphe $G = (V, E)$ contient un 1 en ligne i et en colonne j si l'arête $\{i, j\}$ existe, et un 0 sinon :

$$A_{ij}(G) = \begin{cases} 1 & \text{si } \{i,j\} \in E, \\ 0 & \text{sinon.} \end{cases}$$

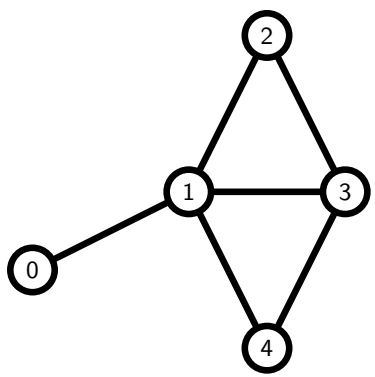
Matrice d'adjacence

Définition 2

La **matrice d'adjacence** $A(G)$ du graphe $G = (V, E)$ contient un 1 en ligne i et en colonne j si l'arête $\{i, j\}$ existe, et un 0 sinon :

$$A_{ij}(G) = \begin{cases} 1 & \text{si } \{i,j\} \in E, \\ 0 & \text{sinon.} \end{cases}$$

Exemple 2



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

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

010010011001010

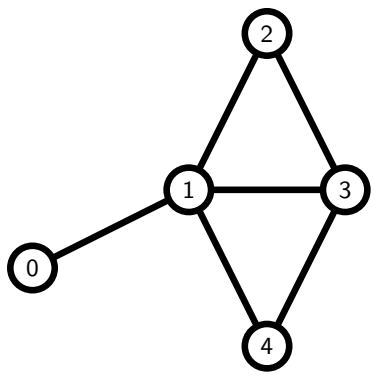
Matrice d'adjacence

Définition 2

La **matrice d'adjacence** $A(G)$ du graphe $G = (V, E)$ contient un 1 en ligne i et en colonne j si l'arête $\{i, j\}$ existe, et un 0 sinon :

$$A_{ij}(G) = \begin{cases} 1 & \text{si } \{i,j\} \in E, \\ 0 & \text{sinon.} \end{cases}$$

Exemple 2



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

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

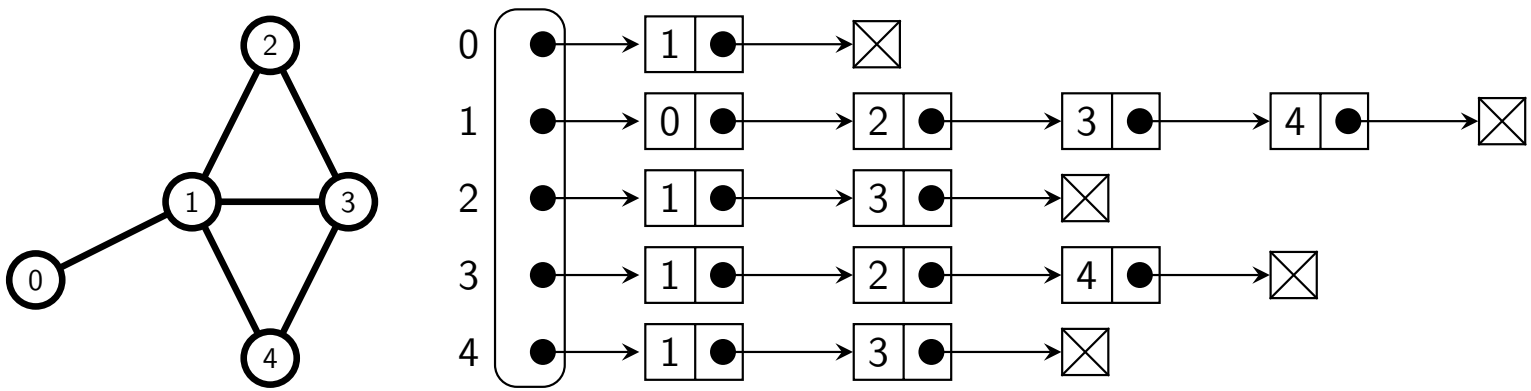
010010011001010

Les connexions sont symétriques, donc $A(G)$ aussi, et on pourrait donc n'en garder que la moitié. S'il n'y a pas de poids, une chaîne binaire suffit.

Listes d'adjacence

Les **listes d'adjacence** encodent, pour chaque sommet du graphe $G = (V, E)$, la liste des voisins de ce sommet.

Exemple 3



Graphviz et dot

Graphviz est une suite d'outils permettant de visualiser des graphes exprimés dans le langage dot. Le format est simple :

Exemple 4

```
graph G {
    # liste de sommets avec leurs propriétés
    0;
    1;
    2;
    # ...
    # liste d'arêtes avec leurs propriétés
    0 -- 1;
    1 -- 2;
    3 -- 7;
    # ...
}
```

Visualisation

Les graphes exprimés dans le langage `dot` peuvent être convertis en images en ligne (`http://www.webgraphviz.com/`) ou à l'aide d'un des outils suivants :

- `dot` : dessine les graphes de manière hiérarchique, typiquement utilisé pour les graphes orientés (voir plus loin) ou les arbres.
- `neato` : utilisé en général pour les graphes de taille raisonnable (moins de 100 sommets).
- `fdp` : similaire à `neato`.
- `sfdp` : plus adapté aux grands graphes.
- `twopi` : dispose un sommet au centre, et les autres sur des cercles concentriques autour de ce centre.
- `circo` : dispose les sommets du graphe sur un cercle.
- `osage` : plus adapté aux graphes "en couches".

Graphes orientés : motivations

- Certains réseaux sont naturellement asymétriques (Twitter, réseaux routiers, graphes de dépendances, généalogies, . . .) ;

Graphes orientés : motivations

- Certains réseaux sont naturellement asymétriques (Twitter, réseaux routiers, graphes de dépendances, généalogies, . . .) ;
- Il faut donc pouvoir représenter des graphes avec des liens asymétriques ;

Graphes orientés : motivations

- Certains réseaux sont naturellement asymétriques (Twitter, réseaux routiers, graphes de dépendances, généalogies, . . .) ;
- Il faut donc pouvoir représenter des graphes avec des liens asymétriques ;
- Ces *graphes orientés* contiennent des *arcs* (u, v) au lieu d'arêtes $\{u, v\}$: la présence d'un lien dans un sens n'implique donc plus la présence de ce lien dans l'autre sens ;

Graphes orientés : motivations

- Certains réseaux sont naturellement asymétriques (Twitter, réseaux routiers, graphes de dépendances, généalogies, . . .) ;
- Il faut donc pouvoir représenter des graphes avec des liens asymétriques ;
- Ces *graphes orientés* contiennent des *arcs* (u, v) au lieu d'arêtes $\{u, v\}$: la présence d'un lien dans un sens n'implique donc plus la présence de ce lien dans l'autre sens ;
- On combinera plus tard poids et orientation.

Concepts de base

Définition 3

Un **graphe orienté** est un couple $G = (V, A)$, où :

- V est un ensemble de **sommets** ;
- $A \subseteq V \times V$ un ensemble d'**arcs** ;

On utilisera aussi les notations $V(G)$ et $A(G)$ pour bien distinguer le graphe G d'un autre graphe.

Concepts de base

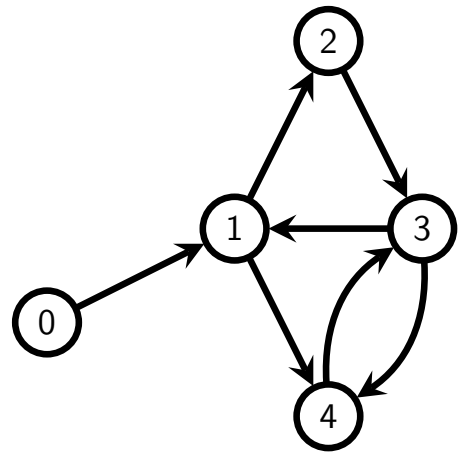
Définition 3

Un **graphe orienté** est un couple $G = (V, A)$, où :

- V est un ensemble de **sommets** ;
- $A \subseteq V \times V$ un ensemble d'**arcs** ;

On utilisera aussi les notations $V(G)$ et $A(G)$ pour bien distinguer le graphe G d'un autre graphe.

Exemple 5



Concepts de base

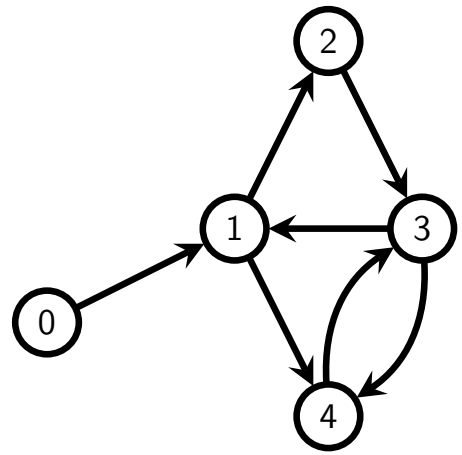
Définition 3

Un **graphe orienté** est un couple $G = (V, A)$, où :

- V est un ensemble de **sommets** ;
- $A \subseteq V \times V$ un ensemble d'**arcs** ;

On utilisera aussi les notations $V(G)$ et $A(G)$ pour bien distinguer le graphe G d'un autre graphe.

Exemple 5



- Un arc (u, v) relie une **source** à une **destination** ; on dit aussi que u est le **prédécesseur** de v, qui est le **successeur** de u. Une boucle est un arc (u, u).

Concepts de base

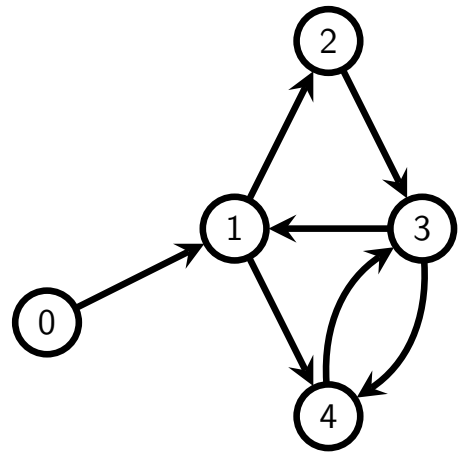
Définition 3

Un **graphe orienté** est un couple $G = (V, A)$, où :

- V est un ensemble de **sommets** ;
- $A \subseteq V \times V$ un ensemble d'**arcs** ;

On utilisera aussi les notations $V(G)$ et $A(G)$ pour bien distinguer le graphe G d'un autre graphe.

Exemple 5



- Un arc (u, v) relie une **source** à une **destination** ; on dit aussi que u est le **prédécesseur** de v, qui est le **successeur** de u. Une boucle est un arc (u, u).
- L'ensemble des **successeurs** d'un sommet v est

$$N_G^+(v) = \{w \mid (v, w) \in A(G)\}$$

L'ensemble des **prédécesseurs** d'un sommet v est

$$N_G^-(v) = \{u \mid (u, v) \in A(G)\}$$

Concepts de base

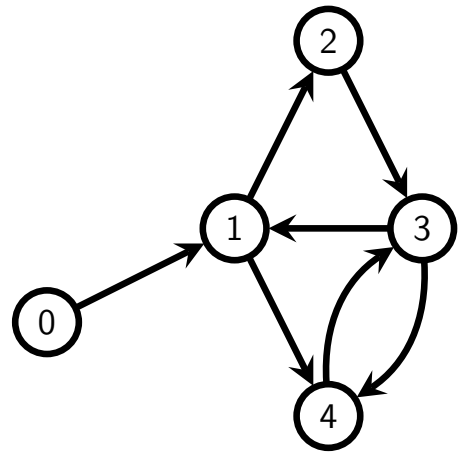
Définition 4

Un **graphe orienté** est un couple $G = (V, A)$, où :

- V est un ensemble de **sommets** ;
- $A \subseteq V \times V$ un ensemble d'**arcs** ;

On utilisera aussi les notations $V(G)$ et $A(G)$ pour bien distinguer le graphe G d'un autre graphe.

Exemple 6



- Le **voisinage** d'un sommet v est l'union de ses prédécesseurs et de ses successeurs ;

Concepts de base

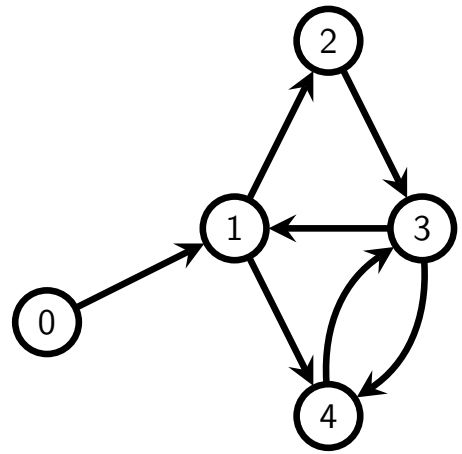
Définition 4

Un **graphe orienté** est un couple $G = (V, A)$, où :

- V est un ensemble de **sommets** ;
- $A \subseteq V \times V$ un ensemble d'**arcs** ;

On utilisera aussi les notations $V(G)$ et $A(G)$ pour bien distinguer le graphe G d'un autre graphe.

Exemple 6



- Le **voisinage** d'un sommet v est l'union de ses prédécesseurs et de ses successeurs ;
- Le **degré entrant** du sommet v est $\deg_G^-(v) = |N_G^-(v)|$;

Concepts de base

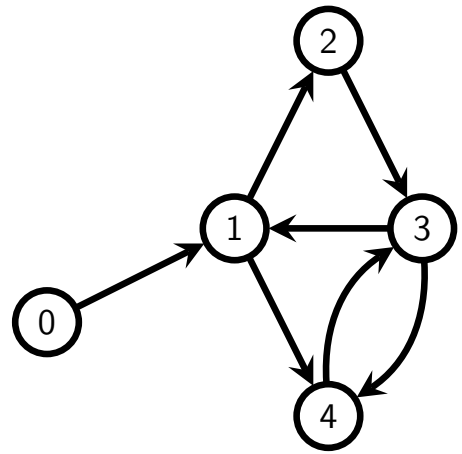
Définition 4

Un **graphe orienté** est un couple $G = (V, A)$, où :

- V est un ensemble de **sommets** ;
- $A \subseteq V \times V$ un ensemble d'**arcs** ;

On utilisera aussi les notations $V(G)$ et $A(G)$ pour bien distinguer le graphe G d'un autre graphe.

Exemple 6



- Le **voisinage** d'un sommet v est l'union de ses prédécesseurs et de ses successeurs ;
- Le **degré entrant** du sommet v est $\deg_G^-(v) = |N_G^-(v)|$;
- Le **degré sortant** du sommet v est $\deg_G^+(v) = |N_G^+(v)|$.

Chemins dans un graphe orienté

- Un **chemin** dans un graphe **orienté** $G = (V, A)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $(u_i, u_{i+1}) \in A$ pour $0 \leq i \leq p - 2$.

  La **longueur du chemin** est p.

Chemins dans un graphe orienté

- Un **chemin** dans un graphe **orienté** $G = (V, A)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $(u_i, u_{i+1}) \in A$ pour $0 \leq i \leq p - 2$.

  La **longueur du chemin** est p.

- Un **chemin simple** dans un graphe **orienté** $G = (V, A)$ est un chemin dont tous les arcs sont distincts.

Chemins dans un graphe orienté

- Un **chemin** dans un graphe **orienté** $G = (V, A)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $(u_i, u_{i+1}) \in A$ pour $0 \leq i \leq p - 2$.

  La **longueur du chemin** est p.

- Un **chemin simple** dans un graphe **orienté** $G = (V, A)$ est un chemin dont tous les arcs sont distincts.

- Un **chemin élémentaire** dans un graphe **orienté** $G = (V, A)$ est un chemin dont tous les sommets sont distincts.

Chemins dans un graphe orienté

- Un **chemin** dans un graphe **orienté** $G = (V, A)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $(u_i, u_{i+1}) \in A$ pour $0 \leq i \leq p - 2$.

  La **longueur du chemin** est p.

- Un **chemin simple** dans un graphe **orienté** $G = (V, A)$ est un chemin dont tous les arcs sont distincts.

- Un **chemin élémentaire** dans un graphe **orienté** $G = (V, A)$ est un chemin dont tous les sommets sont distincts.

- Un **circuit** (on dit aussi **cycle**) dans un graphe **orienté** $G = (V, A)$ est un chemin dont les deux extrémités sont identiques.

Chemins dans un graphe orienté

- Un **chemin** dans un graphe **orienté** $G = (V, A)$ est une séquence de sommets $P = (u_0, u_1, u_2, \ldots, u_{p-1})$, où $(u_i, u_{i+1}) \in A$ pour $0 \leq i \leq p - 2$.

  La **longueur du chemin** est p.

- Un **chemin simple** dans un graphe **orienté** $G = (V, A)$ est un chemin dont tous les arcs sont distincts.

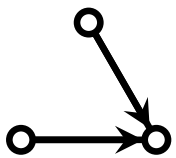
- Un **chemin élémentaire** dans un graphe **orienté** $G = (V, A)$ est un chemin dont tous les sommets sont distincts.

- Un **circuit** (on dit aussi **cycle**) dans un graphe **orienté** $G = (V, A)$ est un chemin dont les deux extrémités sont identiques.

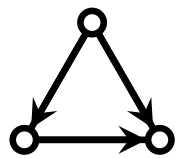
- Un **cycle simple** dans un graphe **orienté** est un circuit qui est un chemin simple.

Chemins dans un graphe orienté

Exemple 7



ceci n'est pas un chemin



ceci n'est pas un cycle

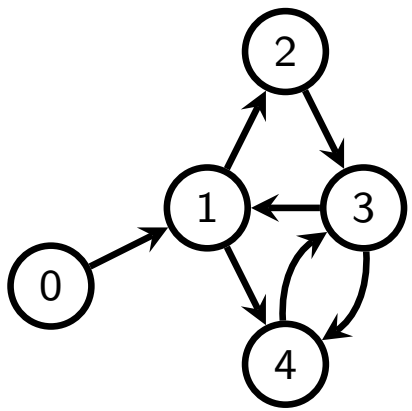
- un **descendant** d'un sommet u dans un graphe orienté G est un sommet $v \neq u$ tel qu'il existe un chemin (**orienté**) de u vers v dans G.

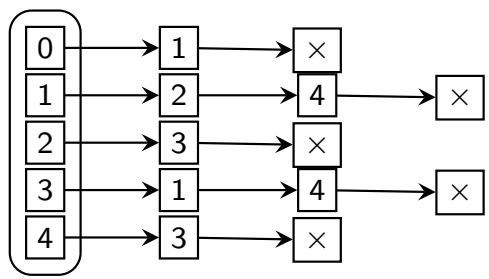
Implémentation des graphes orientés

- Les implémentations du cas non orienté s'adaptent facilement et se simplifient même :
  - matrice d'adjacence : asymétrique, un seul ajout ;
  - listes d'adjacence : successeurs, un seul ajout ;

Exemple 8

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





L'API `GrapheOrienté`

Les algorithmes étudiés supposeront l'existence d'une classe `Graphe` proposant les méthodes suivantes :

Méthode
`ajouter_arc(u, v)`
`ajouter_arcs(séquence)`
`ajouter_sommet(sommet)`
`ajouter_sommets(séquence)`
`arcs()`
`boucles()`
`contient_arc(u, v)`
`contient_sommet(u)`
`degre(sommet)`
`degre_entrant(sommet)`
`degre_sortant(sommet)`
`nombre_arcs()`
`nombre_boucles()`
`nombre_sommets()`

Méthode
`predecesseurs(sommet)`
`retirer_arc(u, v)`
`retirer_arcs(séquence)`
`retirer_sommet(sommet)`
`retirer_sommets(séquence)`
`sommets()`
`sous_graphe_induit(séquence)`
`successeurs(sommet)`
`voisins(sommet)`

Graphes orientés en `dot`

- Pour représenter un graphe orienté en `dot`, on utilise :
  - `digraph` au lieu de `graph` ;
  - `->` pour les arcs (au lieu de `--` pour les arêtes) ;
- On ne peut pas mélanger `--` et `->` : soit le graphe est orienté, soit il ne l'est pas.

Bibliographie

[1] László Babai.

Graph isomorphism in quasipolynomial time [extended abstract].

In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 684–697. ACM, 2016.