Graphes

Plan

Plan

1. Graphes orientés.
2. Graphes non orientés.

Plan

1. Graphes orientés.
2. Graphes non orientés.
3. Graphes valués.

Plan

1. Graphes orientés.
2. Graphes non orientés.
3. Graphes valués.
4. Le problème des plus courts chemins.

Plan

1. Graphes orientés.
2. Graphes non orientés.
3. Graphes valués.
4. Le problème des plus courts chemins.
5. Un algorithme sur les graphes : l'algorithme de Dijkstra.

Graphes orientés : définition

Graphe fini (orienté)

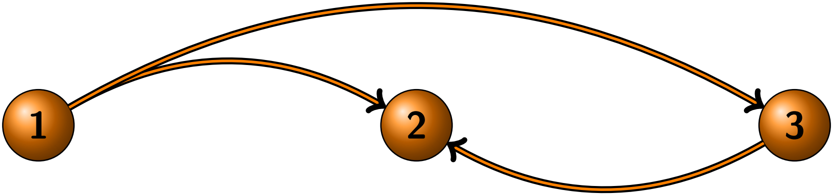
Un **graphe fini (orienté)** G est un couple (S, A) où S est un ensemble fini, et A est une partie de $S \times S$. L'ensemble S est l'ensemble des **sommets** du graphe G et A est l'ensemble des **arêtes** du graphe G.
Remarque : Parfois, on exclut dans la définition de graphe les couples (S, A) où A contient des boucles, c'est-à-dire les éléments de la forme (i, i).

Graphes orientés : exemples

$$G = (\{1, 2, 3\}, \{(1, 2), (1, 3), (3, 2)\})$$

Graphes orientés : exemples

$$G = (\{1,2,3\}, \{(1,2),(1,3),(3,2)\})$$

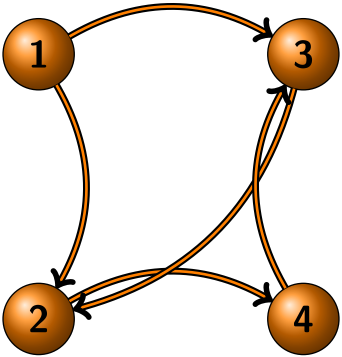


Graphes orientés : exemples

$$G = (\{1, 2, 3, 4\}, \{(1, 2), (1, 3), (3, 2), (2, 4), (4, 3)\})$$

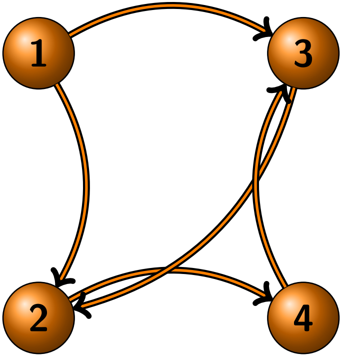
Graphes orientés : exemples

$$G = (\{1,2,3,4\}, \{(1,2),(1,3),(3,2),(2,4),(4,3)\})$$



Graphes orientés : exemples

$$G = (\{1,2,3,4\}, \{(1,2),(1,3),(3,2),(2,4),(4,3)\})$$



Question : combien y-a-t-il de graphes orientés à n sommets sans boucle ?

Graphes non orientés : définition

Graphe non orienté

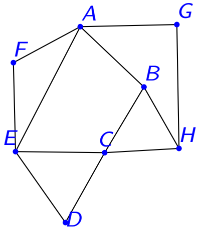
Soit $G = (S, A)$ un graphe fini. On dit que G est non orienté si pour toute arête (i, j) de A, l'élément (j, i) est une arête de G.
Parfois, on définit les graphes non orientés en considérant que l'ensemble des arêtes A est une partie de $\{\{i, j\}, (i, j) \in S\}$

Graphes non orientés : exemples

$$\begin{aligned} G &= (\{A, B, C, D, E, F, G, H\}, \{\{A, B\}, \{A, E\}, \{A, F\}, \{A, G\}, \\ &\quad \{B, C\}, \{B, H\}, \{C, D\}, \{C, E\}, \{C, H\}, \{D, E\}, \{E, F\}, \{G, H\}\}) \end{aligned}$$

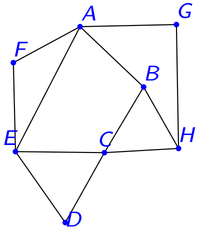
Graphes non orientés : exemples

$$G = (\{A, B, C, D, E, F, G, H\}, \{\{A, B\}, \{A, E\}, \{A, F\}, \{A, G\},
\{B, C\}, \{B, H\}, \{C, D\}, \{C, E\}, \{C, H\}, \{D, E\}, \{E, F\}, \{G, H\}\})$$



Graphes non orientés : exemples

$$\begin{aligned} G &= (\{A, B, C, D, E, F, G, H\}, \{\{A, B\}, \{A, E\}, \{A, F\}, \{A, G\}, \\ &\quad \{B, C\}, \{B, H\}, \{C, D\}, \{C, E\}, \{C, H\}, \{D, E\}, \{E, F\}, \{G, H\}\}) \end{aligned}$$



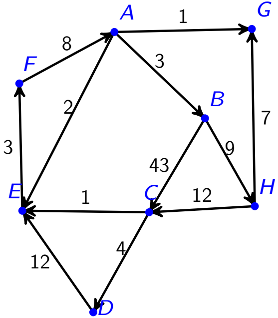
Question : combien y-a-t-il de graphes non orientés à n sommets (sans boucle) ?

Graphes valués : définition

Graphe valué ou à poids

Un graphe valué $G = (S, A, \phi)$ est un graphe (S, A) muni d'une application $\phi : A \to \mathbb{R}$. L'application ϕ est appelée valuation du graphe.

Graphes valués : exemples



Modélisations avec des graphes dans la vie quotidienne :

1. Les réseaux sociaux
2. le plan du métro, une carte routière etc.
3. les probabilités de transition dans un graphe d'états.

Représentations informatiques des graphes

1. matrices d'adjacence

Représentations informatiques des graphes

1. matrices d'adjacence
2. listes d'adjacence.

Exemples

Pour $G = (\{1,2,3,4\}, \{(1,2),(1,3),(3,2),(2,4),(4,3)\})$:

Exemples

Pour $G = (\{1, 2, 3, 4\}, \{(1, 2), (1, 3), (3, 2), (2, 4), (4, 3)\})$:

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

Exemples

Pour $G = (\{1, 2, 3, 4\}, \{(1, 2), (1, 3), (3, 2), (2, 4), (4, 3)\})$:

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

$$L_G = [[2, 3], [4], [2], [3]]$$

Exemples

$$
\begin{aligned}
G \quad = \quad & (\{A, B, C, D, E, F, G, H\}, \{(A, B, 2), (E, A, 6), (A, F, 13), (G, A, 5), \\
& (B, C, 10), (H, B, 11), (C, D, 3), (E, C, 1), \\
& (C, H, 19), (D, E, 1), (E, F, 9), (G, H, 4)\})
\end{aligned}
$$

Exemples

$$G = (\{A, B, C, D, E, F, G, H\}, \{(A, B, 2), (E, A, 6), (A, F, 13), (G, A, 5), (B, C, 10), (H, B, 11), (C, D, 3), (E, C, 1), (C, H, 19), (D, E, 1), (E, F, 9), (G, H, 4)\})$$

matrice d'adjacence de G :

$$M_G = \begin{pmatrix} 0 & 2 & 0 & 0 & 0 & 13 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 & 19 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 6 & 0 & 1 & 0 & 0 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \\ 0 & 11 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Exemples

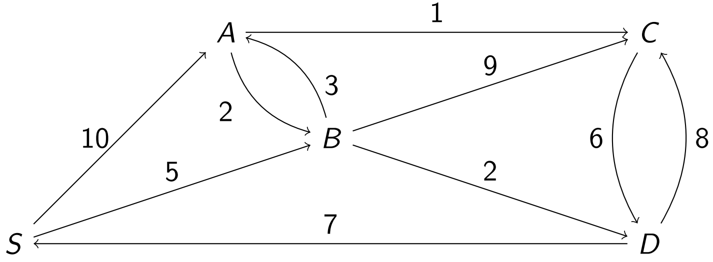
$$
\begin{aligned}
G \;=\; & (\{A, B, C, D, E, F, G, H\}, \{(A, B, 2), (E, A, 6), (A, F, 13), (G, A, 5), \\
& (B, C, 10), (H, B, 11), (C, D, 3), (E, C, 1), \\
& (C, H, 19), (D, E, 1), (E, F, 9), (G, H, 4)\})
\end{aligned}
$$

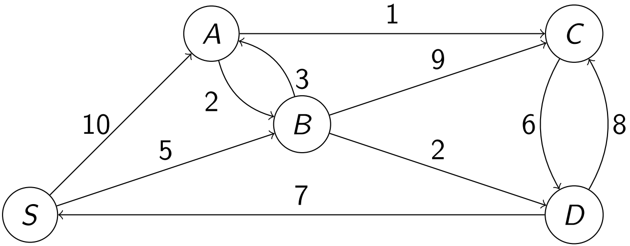
liste d'adjacence de G :

$$
\begin{aligned}
L_G \;=\; & [[(B, 2), (F, 13)], [(C, 10)], [(D, 3), (H, 19)], [(E, 1)], \\
& [(A, 6), (C, 1), (F, 9)], [], [(A, 5), (H, 4)], [(B, 11)]]
\end{aligned}
$$

Le problème des plus courts chemins (le transporteur)

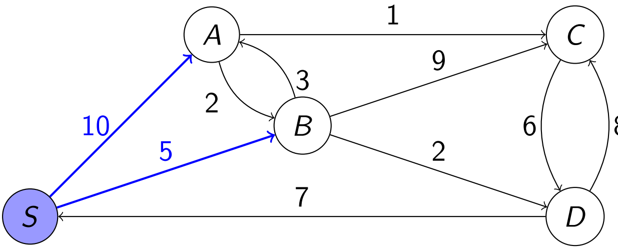
Problème : étant donné un graphe valué, déterminer les plus courts chemins ainsi que le poids de ceux-ci en partant d'un point S donné et en arrivant aux autres points du graphe.
Problème concret correspondant : trouver les chemins optimaux en réseaux ferrés entre Versailles-Chantiers et toutes les autres stations de RER, transiliens et métro. (Comment aller de Versailles-Chantiers à Versailles Rive gauche en marchant le moins possible ?)
Exemple :





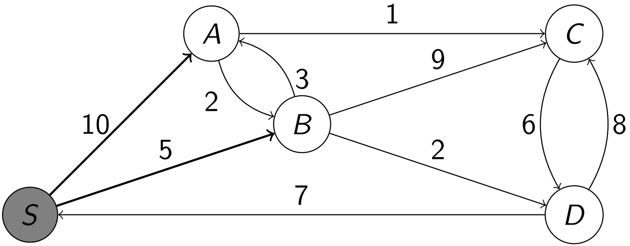
destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	0					[S]
$S \to A$	∞					
$S \to B$	∞					
$S \to C$	∞					
$S \to D$	∞					

Algorithme de Dijkstra : exemple



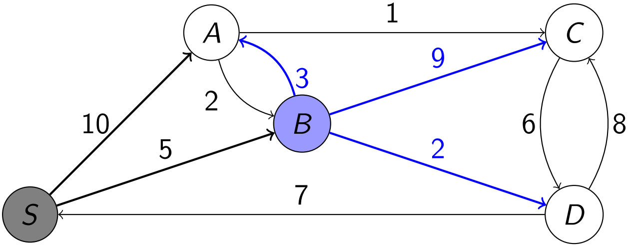
destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	$-$				$[S]$
$S \to A$	∞	10				$[S, A]$
$S \to B$	∞	5				$[S, B]$
$S \to C$	∞	∞				
$S \to D$	∞	∞				

Algorithme de Dijkstra : exemple

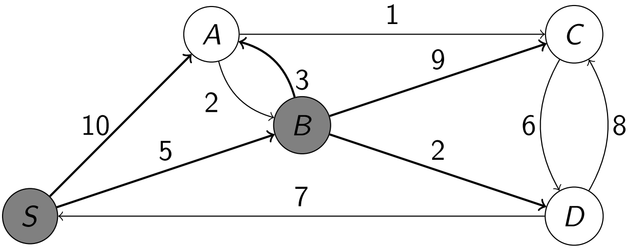


destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	$-$				$\underline{[S]}$
$S \to A$	∞	10				$[S, A]$
$S \to B$	∞	5				$[S, B]$
$S \to C$	∞	∞				
$S \to D$	∞	∞				

Algorithme de Dijkstra : exemple

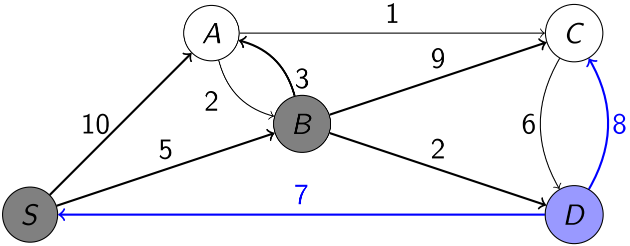


destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	–	–			$[S]$
$S \to A$	∞	10	8			$[S, B, A]$
$S \to B$	∞	$\boxed{5}$	–			$[S, B]$
$S \to C$	∞	∞	14			$[S, B, C]$
$S \to D$	∞	∞	7			$[S, B, D]$



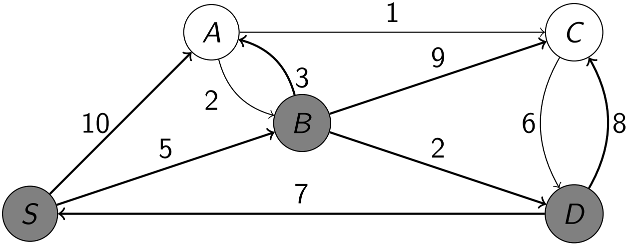
destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	$-$	$-$			$[S]$
$S \to A$	∞	10	8			$[S, B, A]$
$S \to B$	∞	$\boxed{5}$	$-$			$[S, B]$
$S \to C$	∞	∞	14			$[S, B, C]$
$S \to D$	∞	∞	7			$[S, B, D]$

Algorithme de Dijkstra : exemple



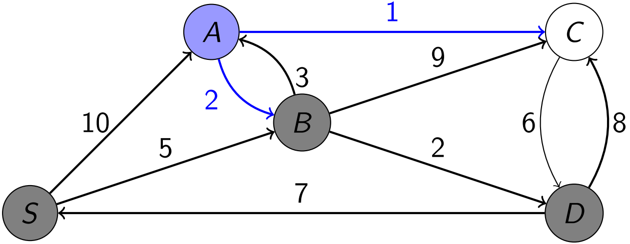
destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	–	–	–		$[S]$
$S \to A$	∞	10	8	8		$[S, B, A]$
$S \to B$	∞	$\boxed{5}$	–	–		$[S, B]$
$S \to C$	∞	∞	14	14		$[S, B, C]$
$S \to D$	∞	∞	$\boxed{7}$	–		$[S, B, D]$

Algorithme de Dijkstra : exemple



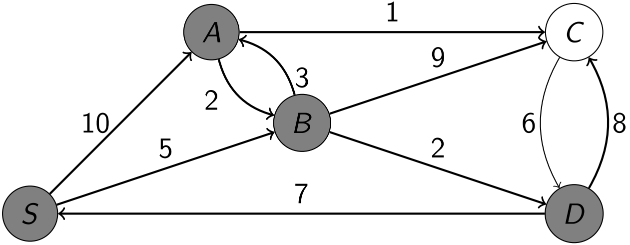
destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	$-$	$-$	$-$		$[S]$
$S \to A$	∞	10	8	8		$[S, B, A]$
$S \to B$	∞	$\boxed{5}$	$-$	$-$		$[S, B]$
$S \to C$	∞	∞	14	14		$[S, B, C]$
$S \to D$	∞	∞	$\boxed{7}$	$-$		$[S, B, D]$

Algorithme de Dijkstra : exemple



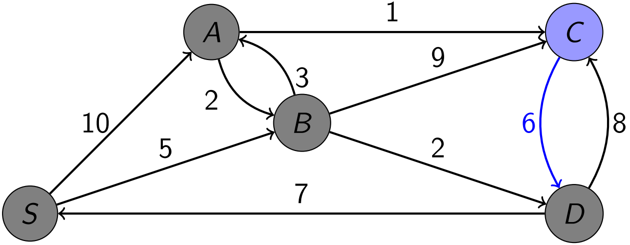
destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	–	–	–	–	$[S]$
$S \to A$	∞	10	8	$\boxed{8}$	–	$[S, B, A]$
$S \to B$	∞	$\boxed{5}$	–	–	–	$[S, B]$
$S \to C$	∞	∞	14	14	9	$[S, B, C]$
$S \to D$	∞	∞	$\boxed{7}$	–	–	$[S, B, D]$

Algorithme de Dijkstra : exemple



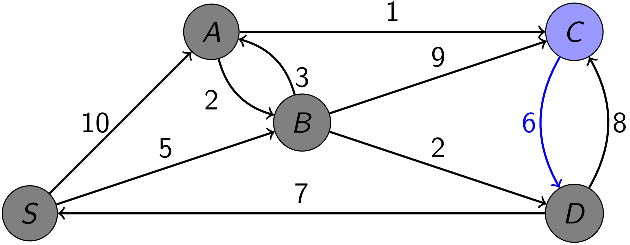
destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	–	–	–	–	$\underline{[S]}$
$S \to A$	∞	10	8	$\boxed{8}$	–	$\underline{[S, B, A]}$
$S \to B$	∞	$\boxed{5}$	–	–	–	$\underline{[S, B]}$
$S \to C$	∞	∞	14	14	9	$[S, B, C]$
$S \to D$	∞	∞	$\boxed{7}$	–	–	$\underline{[S, B, D]}$

Algorithme de Dijkstra : exemple



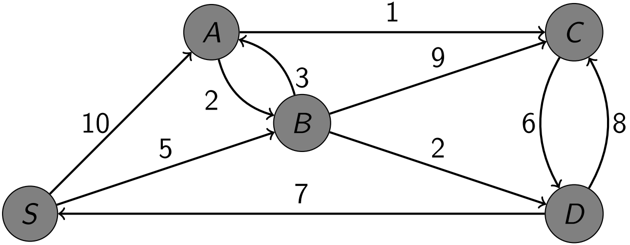
destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	$-$	$-$	$-$	$-$	$[S]$
$S \to A$	∞	10	8	$\boxed{8}$	$-$	$[S, B, A]$
$S \to B$	∞	$\boxed{5}$	$-$	$-$	$-$	$[S, B]$
$S \to C$	∞	∞	14	14	$\boxed{9}$	$[S, B, C]$
$S \to D$	∞	∞	$\boxed{7}$	$-$	$-$	$[S, B, D]$

Algorithme de Dijkstra : exemple



destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	–	–	–	–	$\underline{[S]}$
$S \to A$	∞	10	8	$\boxed{8}$	–	$\underline{[S, B, A]}$
$S \to B$	∞	$\boxed{5}$	–	–	–	$\underline{[S, B]}$
$S \to C$	∞	∞	14	14	$\boxed{9}$	$[S, B, A, C]$
$S \to D$	∞	∞	$\boxed{7}$	–	–	$\underline{[S, B, D]}$

Algorithme de Dijkstra : exemple



destinations	p 0A	p 1A	p 2A	p 3A	p 4A	chemins
$S \to S$	$\boxed{0}$	–	–	–	–	$[S]$
$S \to A$	∞	10	8	$\boxed{8}$	–	$[S, B, A]$
$S \to B$	∞	$\boxed{5}$	–	–	–	$[S, B]$
$S \to C$	∞	∞	14	14	$\boxed{9}$	$[S, B, A, C]$
$S \to D$	∞	∞	$\boxed{7}$	–	–	$[S, B, D]$

Algorithme de Dijkstra : version Python I

```
import numpy as np
def dijkstra(G,source) :
    tab=[[np.Infinity,[]] for i in range(len(G))]
    tab[source]=[0,[source]]
    sommets_a_traiter=[source]
    liste_sommet=[i for i in range(len(G))]
    while sommets_a_traiter!=[] :
        sommet=sommets_a_traiter[0]
        del(sommets_a_traiter[0])
        for j in range(len(G)) :
            if j!=sommet :
                A=tab[sommet][0]+G[sommet][j]
                if tab[j][0]>A :
                    L=tab[sommet][1][:]+[j]
```

Algorithme de Dijkstra : version Python II

```
                tab[j][1]=L[:]
                tab[j][0]=A
    liste_sommet.remove(sommet)
    if liste_sommet!=[] :
        mini_sommet=liste_sommet[0]
        poids=tab[mini_sommet][0]
        for x in liste_sommet :
            if tab[x][0]<poids :
                mini_sommet=x
                poids=tab[x][0]
        sommets_a_traiter.append(mini_sommet)
return tab
```

Algorithme de Dijkstra : version Python avec fonctions auxiliaires

```
import numpy as np
def trouver_mini(liste_sommet,tab) :
    mini_sommet=liste_sommet[0]
    poids=tab[mini_sommet][0]
    for x in liste_sommet :
        if tab[x][0]<poids :
            mini_sommet=x
            poids=tab[x][0]
    return mini_sommet
```

Algorithme de Dijkstra : version Python avec fonctions auxiliaires

```
def relacher_arc(G,sommet,tab) :
    for j in range(len(G)) :
        if j!=sommet :
            A=tab[sommet][0]+G[sommet][j]
            if tab[j][0]>A :
                L=tab[sommet][1][:]+[j]
                tab[j][1]=L[:]
                tab[j][0]=A
    return
```

Algorithme de Dijkstra : version Python avec fonctions auxiliaires

```
def dijkstra(G,source) :
    tab=[[np.Infinity,[]] for i in range(len(G))]
    tab[source]=[0,[source]]
    sommets_a_traiter=[source]
    liste_sommet=[i for i in range(len(G))]
    while sommets_a_traiter!=[] :
        sommet=sommets_a_traiter[0]
        del(sommets_a_traiter[0])
        relacher(G,sommet,tab)
        liste_sommet.remove(sommet)
        if liste_sommet!=[] :
            mini_sommet=trouver_mini(liste_sommet,tab)
            sommets_a_traiter.append(mini_sommet)
    return tab
```

Fin

Des questions ?