LOG 2810: Structures discrètes

Partie 2 : Théorie des Graphes

Julien Dompierre, Philippe Galinier, Robert Roy

Mis à jour par

John Mullins et Fayçal Abouzaid et Foutse Khomh

Ecole Polytechnique de Montréal

Liste des Chapitres

Module Graphes

Introduction aux graphes

Chapitre .1 Introduction et terminologie (5)

- Paire (6) • Couple (7)
- Graphe - Sommet (8)
- Arc et boucle (9)
- Graphe simple (10)
- Multigraphe (11)
- Pseudographe (12)
- Graphe orienté (13)
- Multigraphe orienté (14)

Suite page suivante...

Module

... suite de la page précédente

Chapitre .2 Vocabulaire et exemples de graphes (16)

- Sommets adjacents (16) • Degré d'un sommet (17)
- Lemme des poignées de mains (19)
- Degrés intérieur et extérieur (22)
- Graphe complet (24)
- Cycle (25) • Roue (26) • Cube (27)
- Graphe biparti (28) • Graphe biparti complet (37)
- Sous-graphe (38) • Union de graphes (39)
- Graphe régulier (40)
- Réseau (41)

Graphes: Exemple introductif

Le site web de la Société des transports de la communauté urbaine de Montréal

`www.stcum.qc.ca`

En particulier l'application *Tous azimuts*

`www.stcum.qc.ca/azimuts/index.htm`

Cette application est faite en collaboration avec le groupe MADITUC (Modèle d'Analyse Désagrégée des Itinéraires de Transport Urbain Collectif) de l'École Polytechnique

`www.madituc.polymtl.ca`

Définition: une paire

Soit V un ensemble et u et v deux éléments appartenant à cet ensemble. Une **paire** dans V est un ensemble de deux éléments de V. Autrement dit, une paire dans V est un ensemble $\{u, v\}$ tel que $u, v \in V$.

Remarques:

- Les deux paires $\{u, v\}$ et $\{v, u\}$ sont identiques, c'est-à-dire que l'ordre des éléments dans l'ensemble n'a pas d'importance.
- La paire $\{u, v\}$ implique implicitement que $u \neq v$.
- La paire $\{u, u\}$ s'écrit comme un singleton $\{u\}$.

Définition: un couple

Le **couple** (u, v) est la collection *ordonnée* admettant u comme premier élément et v comme deuxième élément.

Soit V un ensemble. Le **produit cartésien** dans V, noté $V \times V$, est

$$V \times V = \{(u, v) \,|\, u \in V \wedge v \in V\}.$$

Remarques:

- Deux couples sont égaux si et seulement si les éléments correspondants sont égaux.
- Il y a un ordre, c'est-à-dire que $(u, v) \neq (v, u)$.
- Le couple (u, u) est un élément de l'ensemble $V \times V$.

Définition: graphe - sommet

Un **graphe** $G = (V, E)$ est constitué d'un ensemble V de sommets et d'un ensemble E d'arcs reliant ces sommets deux à deux.

L'ensemble des sommets d'un graphe est noté V pour *vertices*.

L'ensemble des sommets V est non vide car s'il est vide, alors il n'y a pas de graphe.

Définition: arcs

Les sommets sont reliés deux à deux par un ensemble E, éventuellement vide, d'arcs.

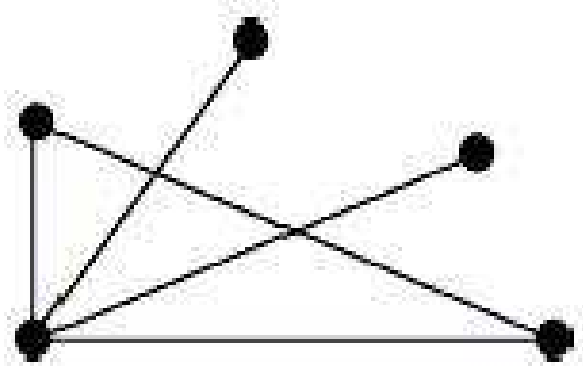
- L'ensemble des arcs est noté E pour *Edges*.
- Un arc entre les sommets u et v peut être **bidirectionnel** (ou **non orienté**) et est défini par la *paire* $\{u, v\}$.
- Un arc entre les sommets u et v peut être **directionnel** (ou **orienté**) et est défini par le *couple* (u, v).
- On peut permettre ou non plusieurs arcs entre les deux mêmes sommets.
- On peut permettre ou non les arcs entre un sommet et lui-même (une **boucle**).
- Type de graphes : simple, multigraphe, pseudographe, graphe orienté et multigraphe orienté.

Définition: graphe simple

Un **graphe simple** $G = (V, E)$ est constitué d'un ensemble non vide V de sommets et d'un ensemble E d'arcs formés de *paires* d'éléments distincts de V.

Remarques:

- $E = \{\{u, v\} \mid u, v \in V \wedge u \neq v\}$.
- Les arcs sont non orientés.
- Une graphe simple est aussi appelé un **graphe non orienté**.
- Il ne peut pas y avoir plus d'un arc entre les deux mêmes sommets.
- Il ne peut pas y avoir de boucle.

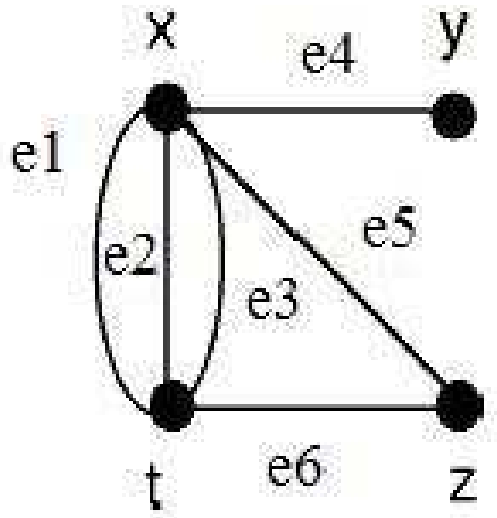


Définition: multigraphe

Un **multigraphe** $G = (V, E)$ est constitué d'un ensemble non vide V de sommets, d'un ensemble E d'arcs formés de *paires* d'éléments distincts de V et d'une fonction f de E dans $\{\{u, v\} \mid u, v \in V \wedge u \neq v\}$. Les arcs e_1 et e_2 sont appelés des **arcs multiples** (ou **arcs parallèles**) si $f(e_1) = f(e_2)$.

Remarques:

- Les arcs sont non orientés.
- Il peut y avoir plus d'un arc entre les deux mêmes sommets.
- Il ne peut pas y avoir de boucle.

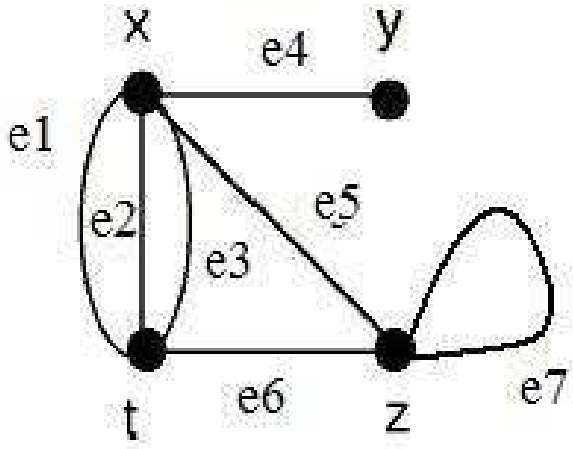


Définition: pseudographe

Un **pseudographe** $G = (V, E)$ est constitué d'un ensemble non vide V de sommets, d'un ensemble E d'arcs formés de *paires* d'éléments de V et d'une fonction f de E dans $\{\{u, v\} \mid u, v \in V\}$. Un arc est une boucle si $f(e) = \{u, u\} = \{u\}$ pour certains $u \in V$.

Remarques:

- Les arcs sont non orientés.
- Il peut y avoir plus d'un arc entre les deux mêmes sommets.
- Il peut y avoir des boucles.



Définition: graphe orienté

Un **graphe orienté** $G = (V, E)$ est constitué d'un ensemble non vide V de sommets et d'un ensemble E d'arcs formés de *couples* d'éléments de V.

Remarques:

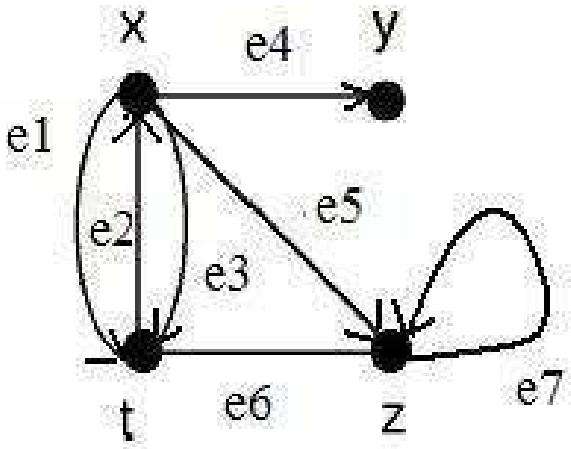
- $E = \{(u, v) \mid u, v \in V\} \subseteq V \times V$.
- Les arcs sont orientés.
- Il ne peut pas y avoir d'arcs multiples de même direction entre les deux mêmes sommets.
- Il peut y avoir des boucles.

Définition: multigraphe orienté

Un **multigraphe orienté** $G = (V, E)$ est constitué d'un ensemble non vide V de sommets, d'un ensemble E d'arcs formés de *couples* d'éléments de V et d'une fonction f de E dans $\{(u, v) \mid u, v \in V\}$. Les arcs e_1 et e_2 sont appelés des **arcs multiples** si $f(e_1) = f(e_2)$.

Remarques:

- Les arcs sont orientés.
- Il peut y avoir des arcs multiples de même direction entre les deux mêmes sommets.
- Il peut y avoir des boucles.



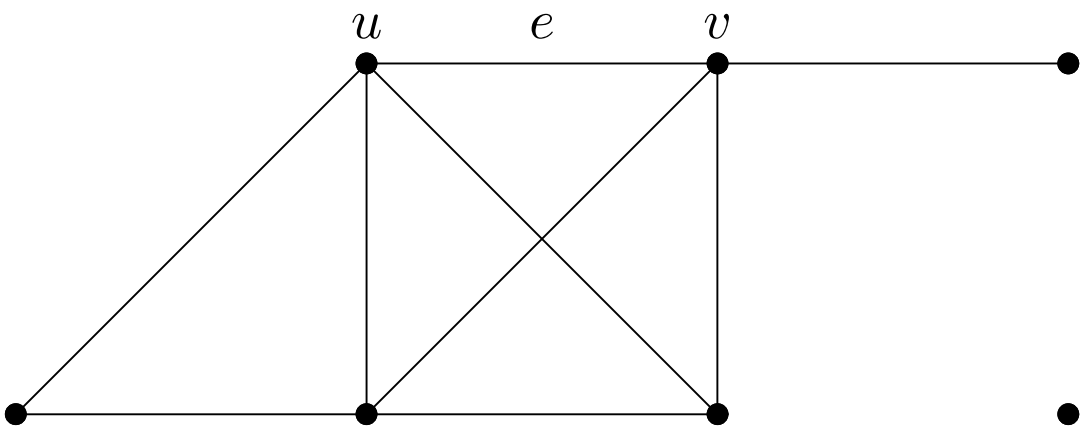
Terminologie des graphes

Type	Arcs	Arcs multiples?	Boucles?
Graphe simple	Non orientés	non	non
Multigraphe	Non orientés	oui	non
Pseudographe	Non orientés	oui	oui
Graphe orienté	Orientés	non	oui
Multigraphe orienté	Orientés	oui	oui

Note: Le terme générique **graphe** décrit des graphes avec des arcs orientés ou non, des boucles ou non et des arcs multiples ou non.

Définition: sommets adjacents

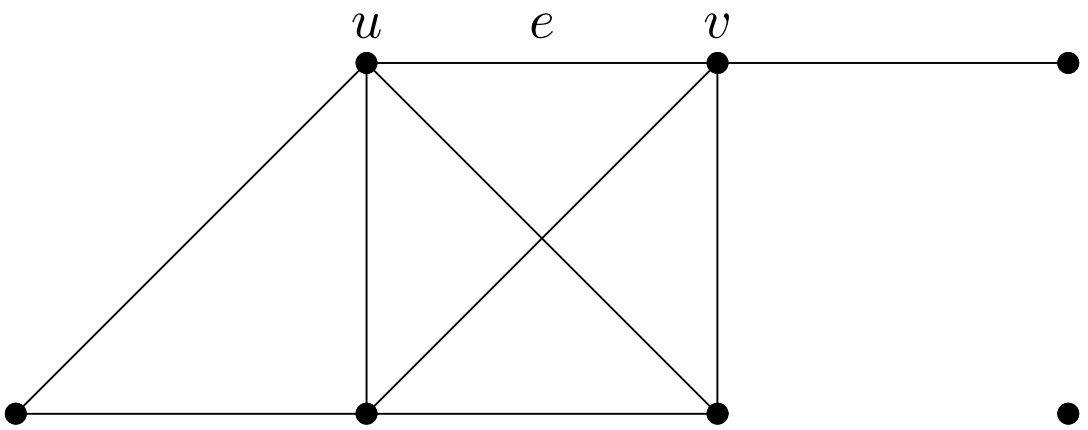
Deux sommets u et v dans un graphe non orienté G sont **adjacents** (ou **voisins**) dans G si $\{u, v\}$ est un arc de G.
Si $e = \{u, v\}$, l'arc e est **incident** aux sommets u et v.
On dit également que l'arc e **relie** u et v.
Les sommets u et v sont les **points terminaux** de l'arc $\{u, v\}$.



Définition: degré d'un sommet

Le **degré** d'un sommet dans un graphe non orienté est le nombre d'arcs incidents à ce sommet, et une boucle sur un sommet contribue deux fois au degré du sommet.

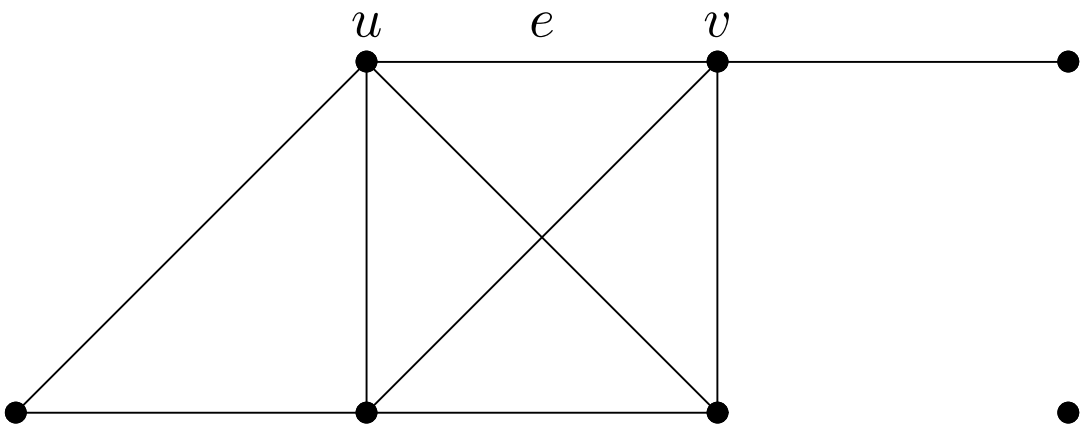
Le degré du sommet v est noté $\deg(v)$.



Définition: sommets isolés et pendants

Un sommet de degré 0 est un sommet **isolé**. Il s'ensuit qu'un sommet isolé n'est adjacent à aucun autre sommet.

Un sommet est **pendant** si et seulement si il est de degré 1. Il s'ensuit qu'un sommet pendant est adjacent à un seul autre sommet.



Théorème des poignées de mains

THÉORÈME: Soit $G = (V, E)$ un graphe non orienté avec e arcs. Alors

$$2e = \sum_{v \in V} \deg(v).$$

À noter que ce théorème s'applique même dans le cas d'arcs multiples et de boucles.

PREUVE : Chaque arc contribue 2 fois à la somme des sommets puisque chaque arc est adjacent avec exactement 2 sommets (éventuellement identiques). D'où le résultat.

Sommets de degrés impairs

THÉORÈME: Un graphe non orienté a un nombre pair de sommets de degrés impairs.

PREUVE : Soit e le nombre d'arcs et soient V_1 et V_2 les ensembles de sommets de degré respectifs pair et impairs. Alors

$$2e = \sum_{v \in V_1} deg(v) + \sum_{v \in V_2} deg(v)$$

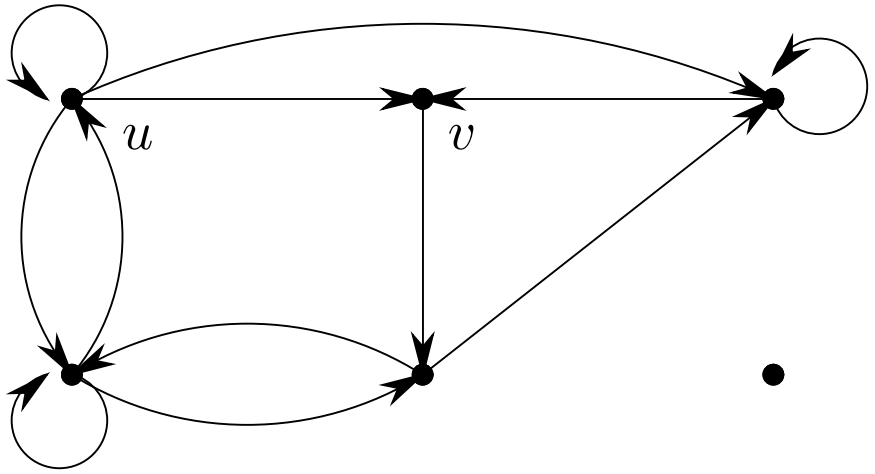
$deg(v)$ est pair pour $v \in V_1$ et donc $deg(v)$ est aussi pair pour $v \in V_2$ puisque la somme est paire. Or tous les termes de cette somme sont impairs, on en a donc un nombre pair.

Définition: sommet adjacent

Quand (u, v) est un arc du graphe orienté G, u est **adjacent** à v et v est **adjacent** à u.

Le sommet u est l'**extrémité initiale** de (u, v) et v est l'**extrémité terminale** ou **finale** de (u, v).

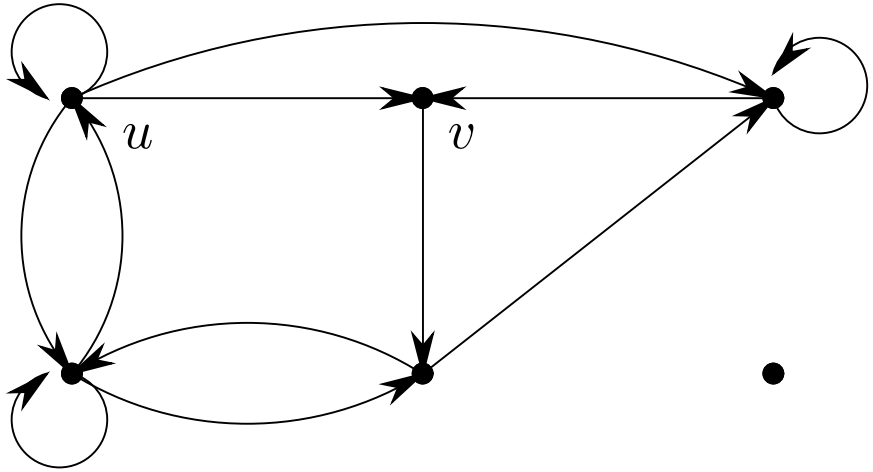
Remarque: Les extrémités initiale et finale d'une boucle sont identiques.



Définition: degrés intérieur et extérieur

Dans un graphe orienté, le **degré intérieur** d'un sommet v, noté $\deg^-(v)$, est le nombre d'arcs qui ont v comme extrémité finale.
Le **degré extérieur** d'un sommet v, noté $\deg^+(v)$, est le nombre d'arcs qui ont v comme extrémité initiale.

Remarque: Une boucle sur un sommet contribue pour 1 à la fois au degré intérieur et au degré extérieur de ce sommet.



Somme de degrés intérieur et extérieur

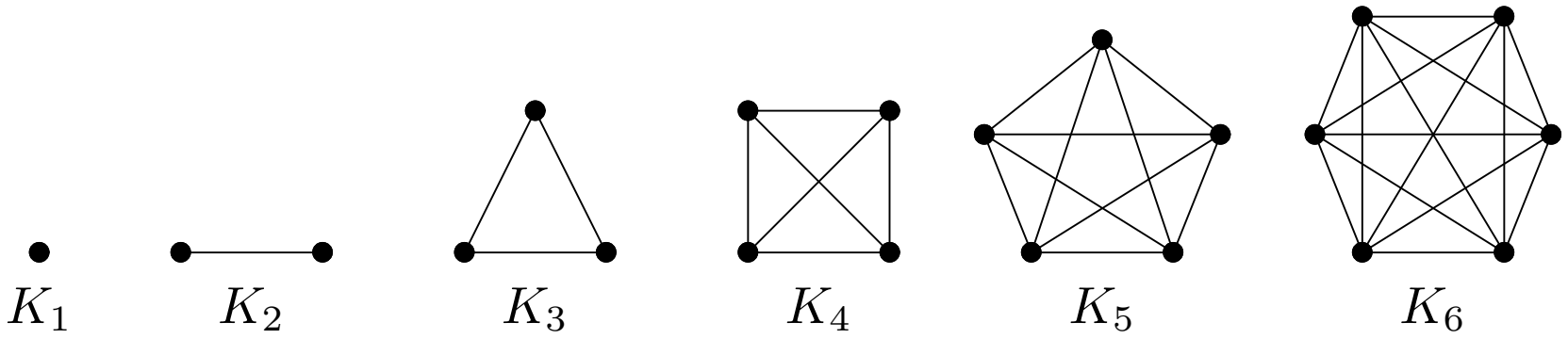
THÉORÈME: Soit $G = (V, E)$ un graphe orienté. Alors

$$\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = |E|.$$

PREUVE : Puisque chaque arc a une extrémité initiale et une finale, les sommes des degrés intérieurs et extérieurs sont identiques et égales au nombre d'arcs du graphe.

Définition: graphe complet

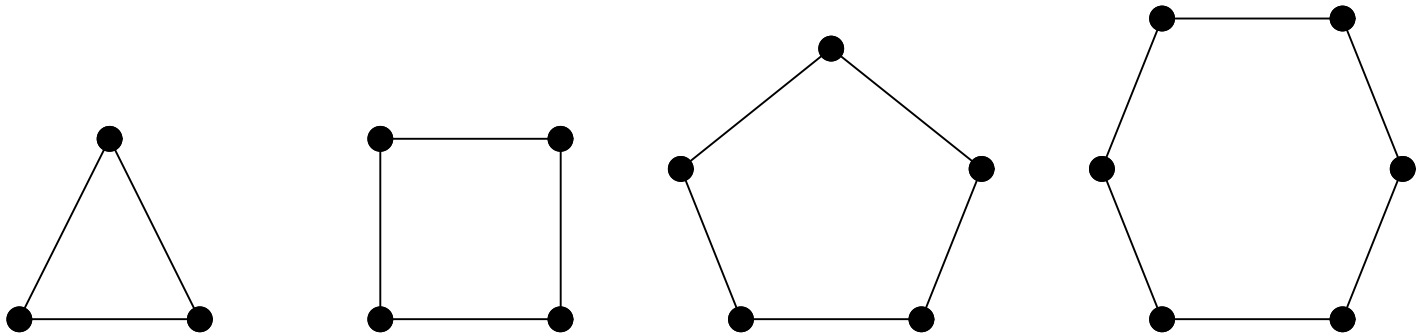
Le **graphe complet** de n sommets noté K_n est le graphe simple qui contient exactement un arc entre chaque paire de sommets distincts.



Graphes K_n pour $1 \leq n \leq 6$

Définition: cycle

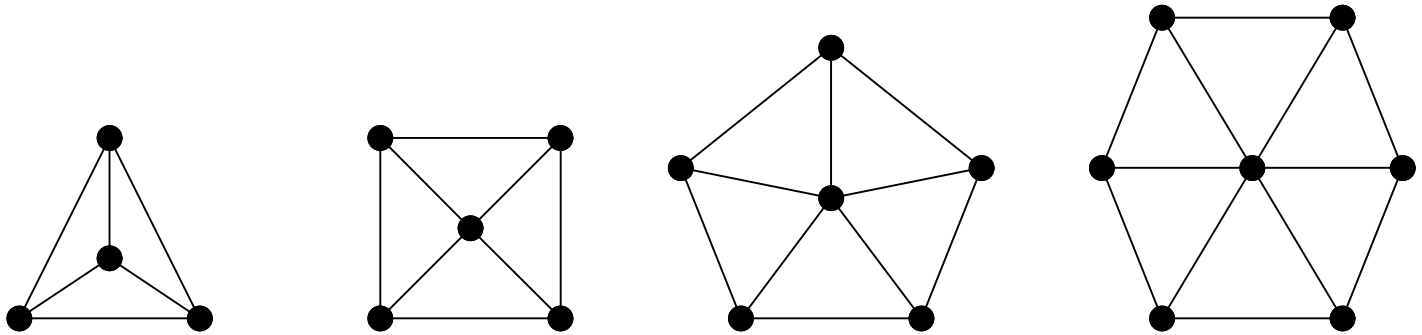
Le **cycle** C_n, pour $n \geq 3$, consiste en n sommets $v_1, v_2, ..., v_n$ et les arcs $\{v_1, v_2\}$, $\{v_2, v_3\}$, ..., $\{v_{n-1}, v_n\}$ et $\{v_n, v_1\}$.



Graphes C_n pour $3 \leq n \leq 6$

Définition: roue

On obtient la **roue** W_n quand on ajoute un sommet supplémentaire au cycle C_n pour $n \geq 3$ et qu'on relie ce nouveau sommet à chacun des sommets de C_n au moyen de nouveaux arcs.

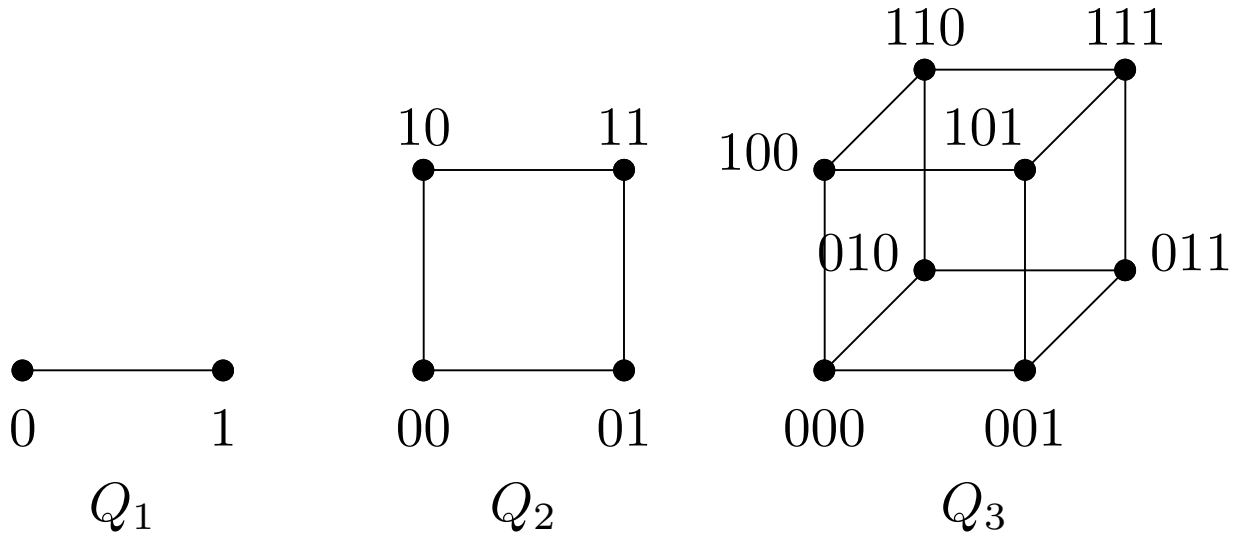


C_3 C_4 C_5 C_6

Graphes W_n pour $3 \leq n \leq 6$

Définition: cube de dimension n

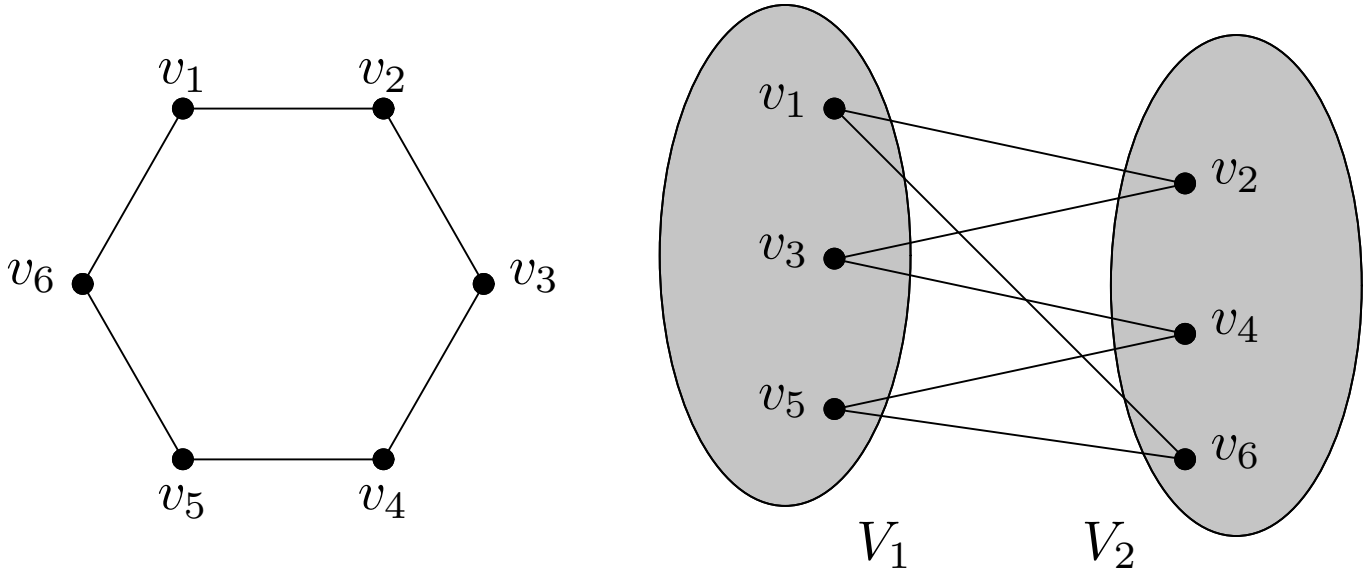
Le **cube de dimension** n noté Q_n est le graphe qui a des sommets représentant les 2^n chaînes binaires de longueur n. Deux sommets sont adjacents si et seulement si les chaînes binaires qu'ils représentent diffèrent d'exactement un bit.



Graphes Q_n pour $1 \leq n \leq 3$

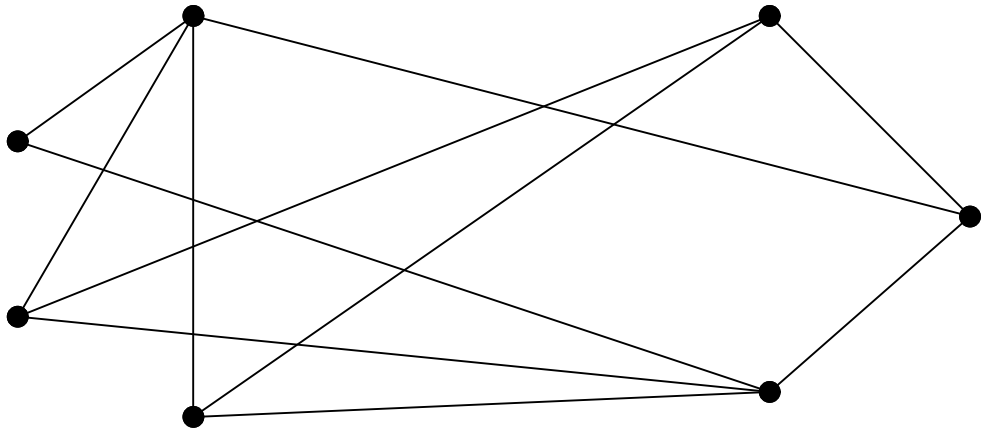
Définition: graphe biparti

Un graphe simple G est **biparti** si l'ensemble V de ses sommets peut être partitionné en deux ensembles non vides et disjoints V_1 et V_2 de telle façon que chaque arc du graphe relie un sommet de V_1 à un sommet de V_2.



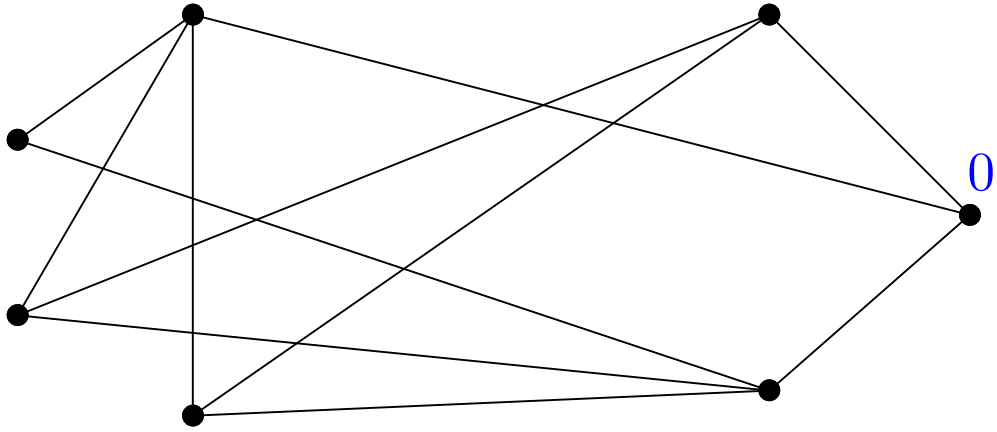
Le cycle C_6 est un graphe biparti

Ce graphe est-il biparti?



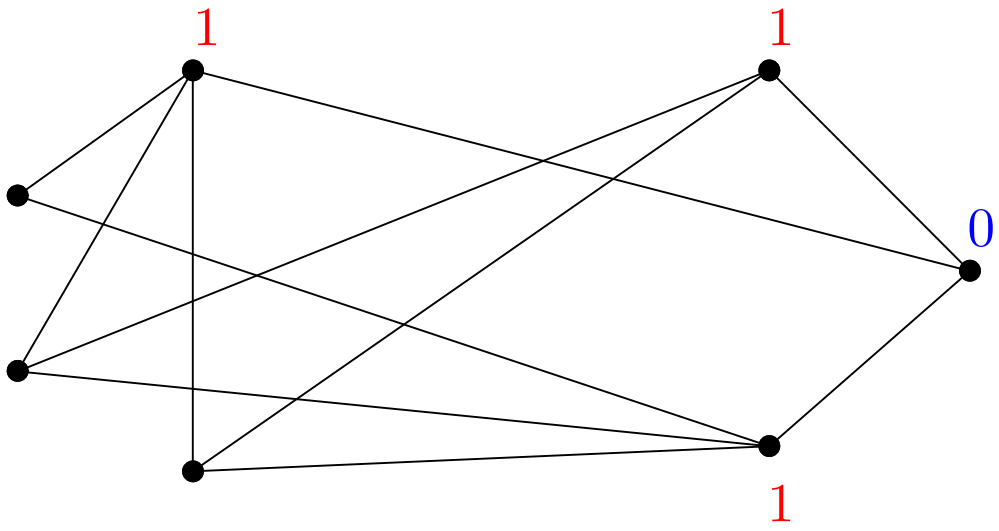
Réponse: Étape 1 de 3

On étiquette un sommet quelconque du graphe avec un 0.



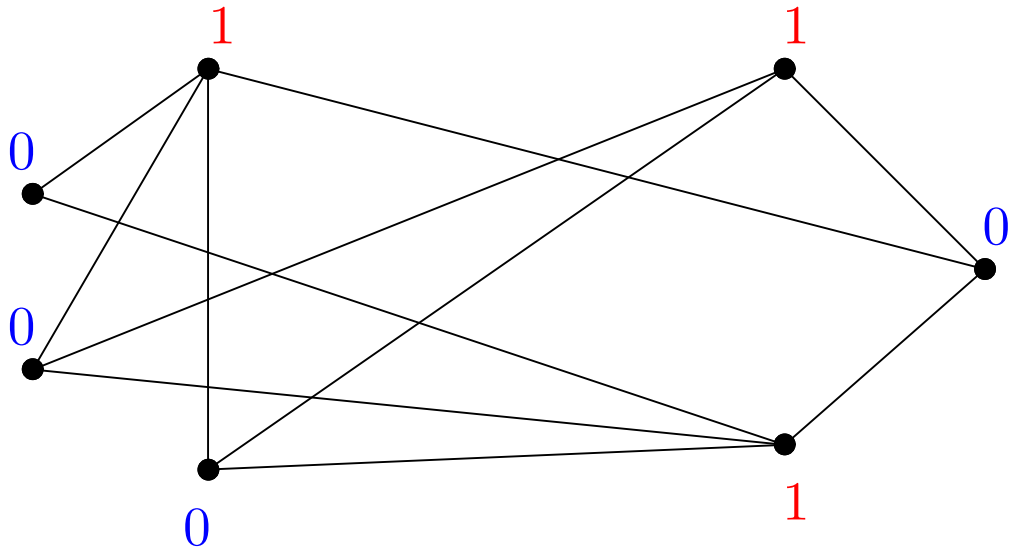
Réponse: Étape 2 de 3

On étiquette les sommets adjacents du premier sommet avec un 1.



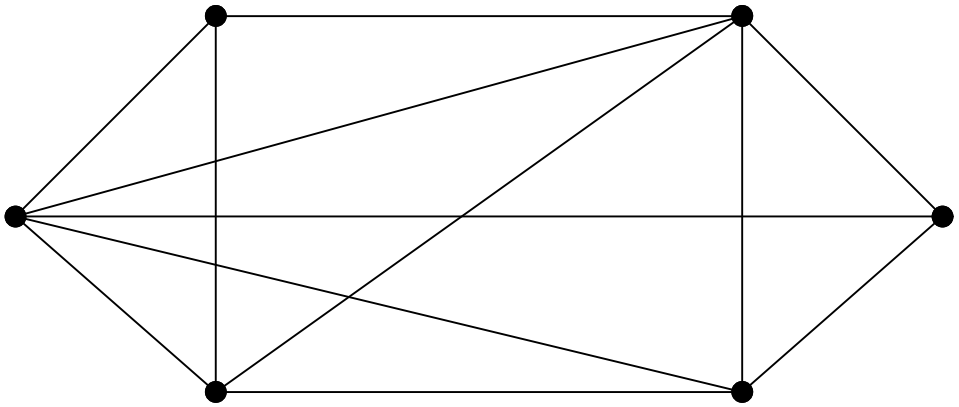
Réponse: Étape 3 de 3

On étiquette les sommets adjacents des sommets adjacents du premier sommet avec un 0.



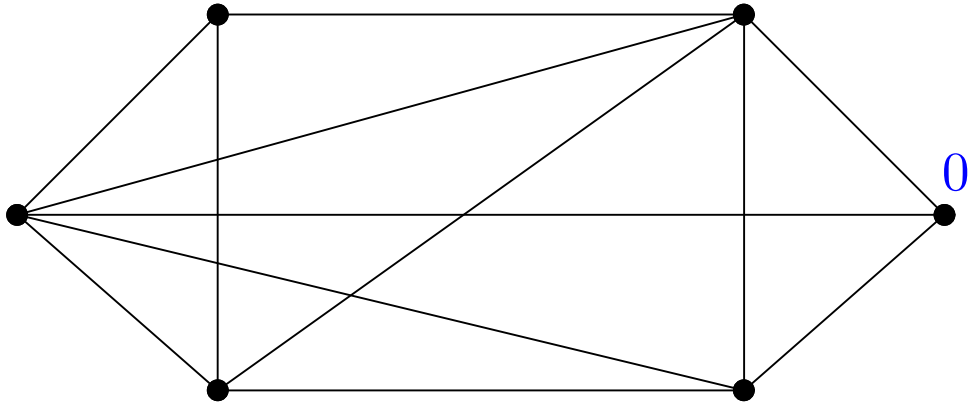
À la fin du processus, si tous les sommets ont une étiquette unique, alors le graphe est biparti.

Ce graphe est-il biparti?



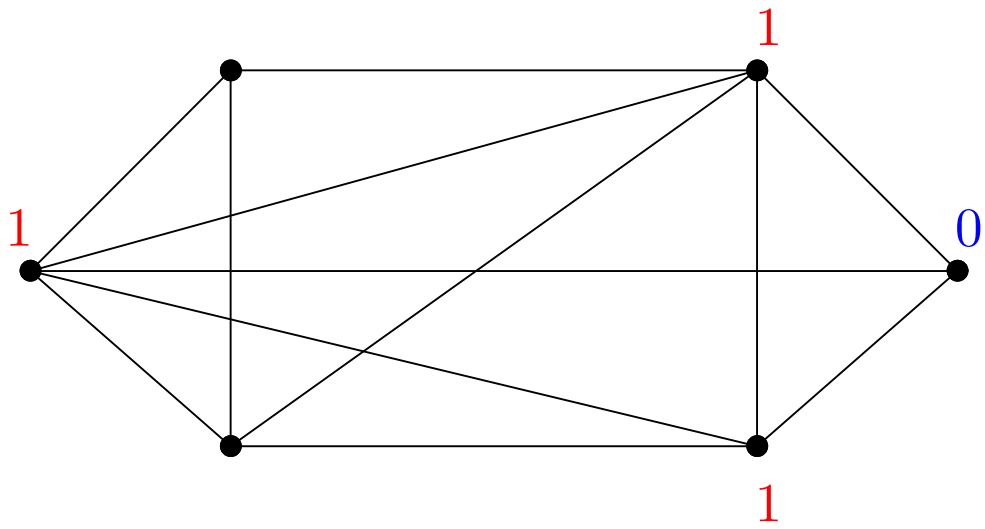
Réponse: Étape 1 de 3

On étiquette un sommet quelconque du graphe avec un 0.



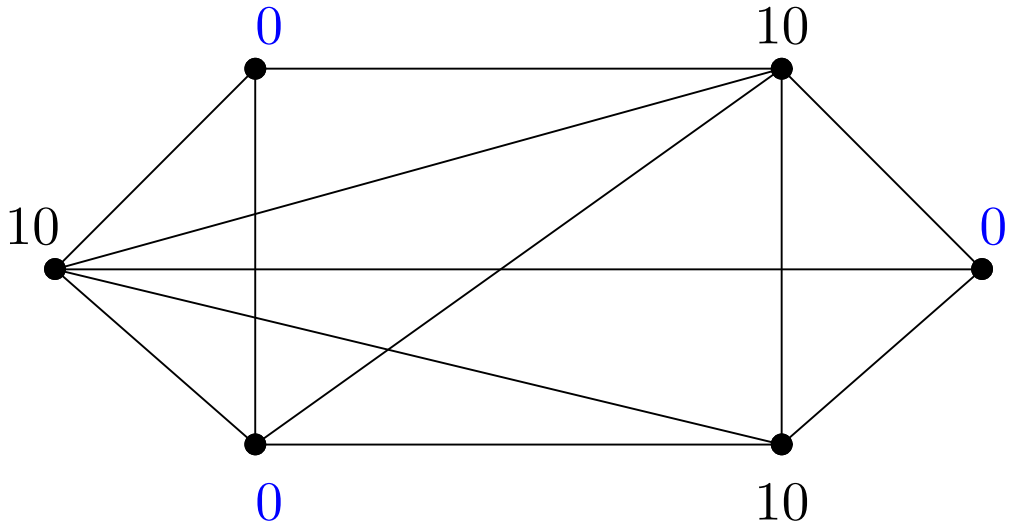
Réponse: Étape 2 de 3

On étiquette les sommets adjacents du premier sommet avec un 1.



Réponse: Étape 3 de 3

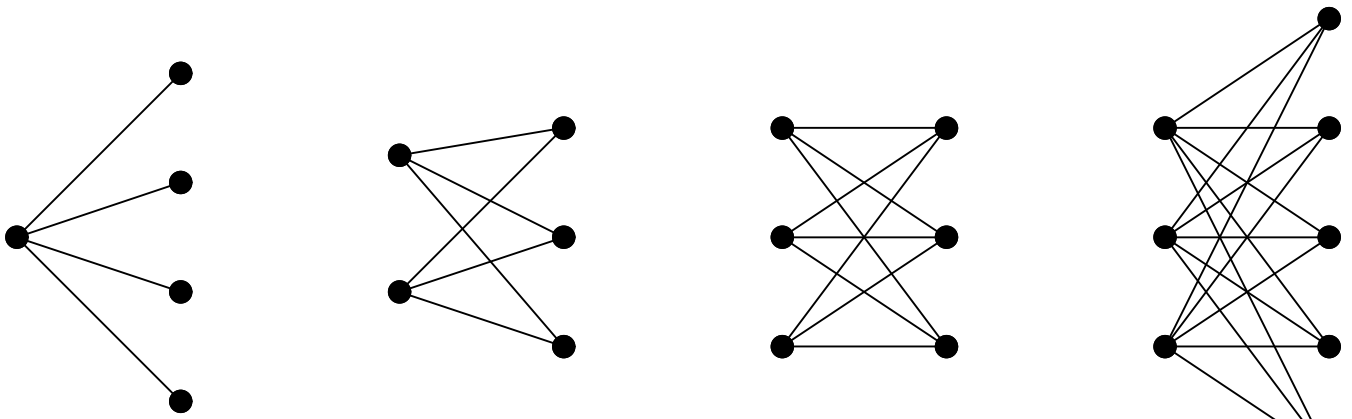
On étiquette les sommets adjacents des sommets adjacents du premier sommet avec un 0.



Dès qu'un sommet est affecté de deux étiquettes différentes, alors le graphe n'est pas biparti.

Définition: graphe biparti complet

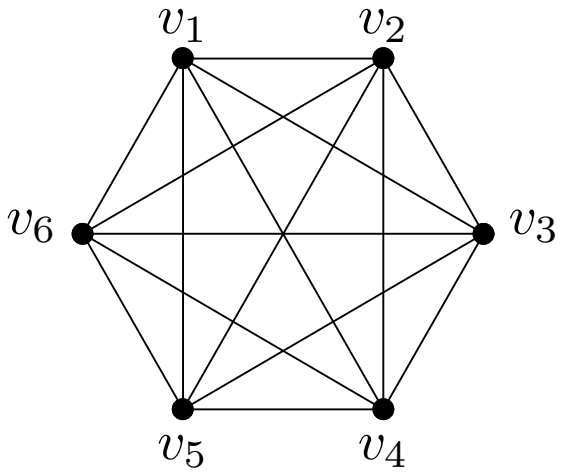
Le **graphe biparti complet** $K_{m,n}$ est un graphe dont l'ensemble des sommets est partitionné en deux sous-ensembles qui ont respectivement m et n sommets. Il y a un arc entre deux sommets si et seulement si un sommet est dans le premier sous ensemble et que l'autre sommet est dans le second sous-ensemble.



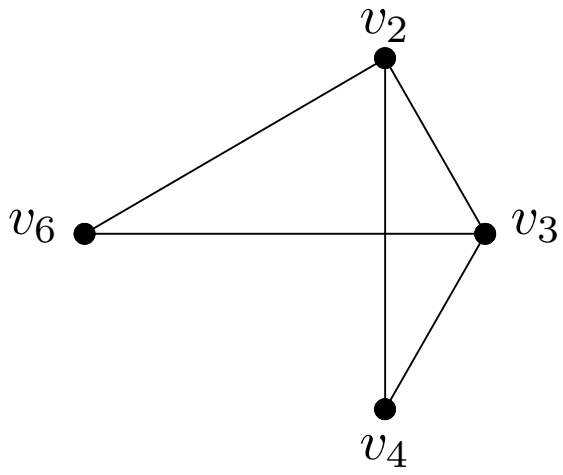
$K_{1,4}$ $K_{2,3}$ $K_{3,3}$ $K_{3,5}$

Définition: sous-graphe

Un **sous-graphe** du graphe $G = (V, E)$ est un graphe $H = (W, F)$ où $W \subseteq V$ et $F \subseteq E$.



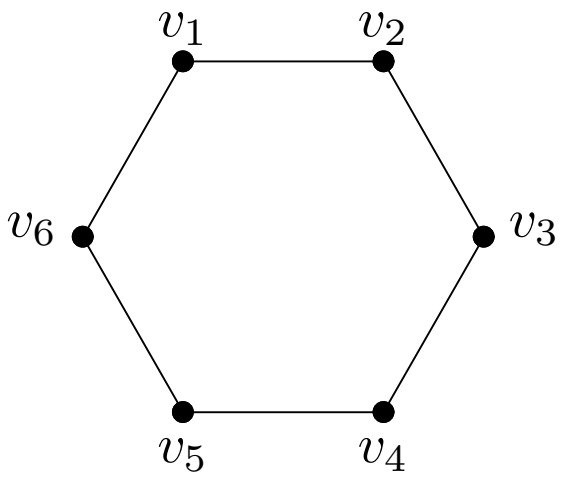
G



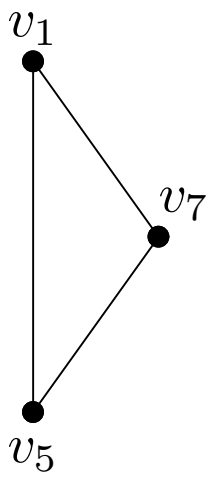
H

Définition: union de graphes

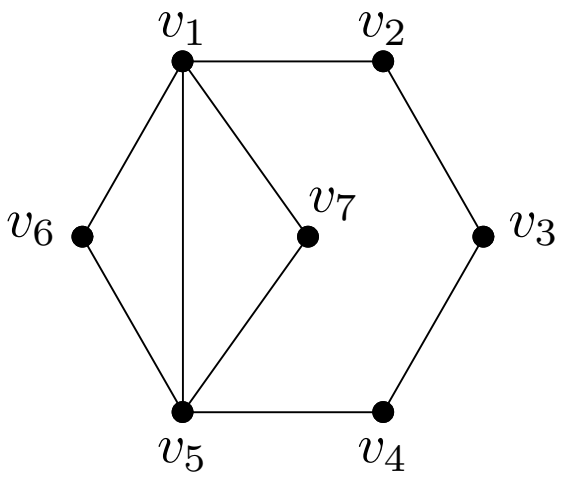
L'**union** de deux graphes simples $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$ est un graphe simple qui contient l'ensemble des sommets $V_1 \cup V_2$ et l'ensemble des arcs $E_1 \cup E_2$. L'union de G_1 et G_2 est notée $G_1 \cup G_2$.



G_1



G_2



$G_1 \cup G_2$

Définition: graphe régulier

Un graphe simple est **régulier** si tous ses sommets sont de degré identique.

Un graphe régulier est **régulier de degré** n si tous ses sommets sont de degré n.

Application: Réseaux locaux

Les ordinateurs et périphériques d'un édifice sont reliés au moyen d'un *réseau local*. Certains types de réseaux locaux sont construits selon une *topologie en étoile*, une *topologie en anneau* ou une *topologie hybride*.

Un réseau local peut être représenté en utilisant un graphe.

Type de réseau	Graphe
topologie en étoile	graphe biparti complet $K_{1,n}$
topologie en anneau	cycle C_n
topologie hybride	roue W_n

d'interconnexion d'ordinateurs parallèles

En mode de traitement simultané (ordinateur parallèle), un processeur a besoin des sorties provenant d'un autre processeur. Les processeurs ont besoin d'être interconnectés par des liens bidirectionnels. On utilise un graphe pour représenter le réseau d'interconnexion.

Application: Réseaux d'interconnexion (suite)

Un réseau d'interconnexion peut être représenté en utilisant un graphe.

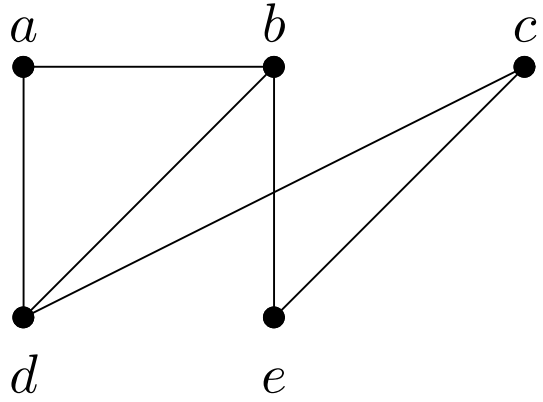
Type de réseau	Graphe	Nb proc	degré proc
réseau complet	complet K_n	n	$n-1$
tableau unidimensionnel	"chaîne"	n	2
réseau maillé (tableau bidimensionnel)	"grille"	n^2	4
hypercube	cube Q_n	2^n	n

Module Isomorphismes

Représentation et isomorphisme de graphes

Représentation par énumération

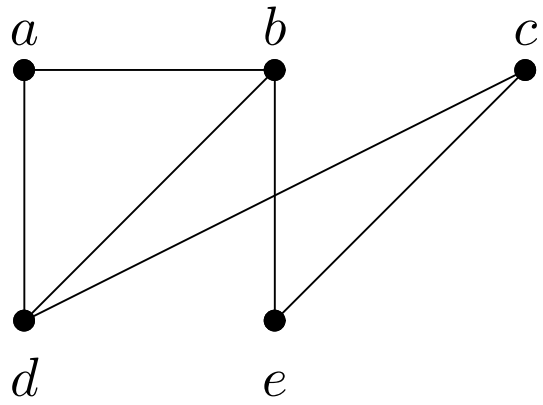
Une manière de représenter un graphe sans arcs multiples est d'énumérer tous les arcs de ce graphe.



$G = (V, E)$ avec $V = \{a, b, c, d, e\}$ et
$E = \{\{a, b\}, \{a, d\}, \{b, d\}, \{b, e\}, \{d, c\}, \{e, c\}\}$.

Représentation par une liste d'adjacence

Une manière de représenter un graphe sans arcs multiples est d'utiliser des **listes d'adjacence** qui spécifient les sommets adjacents à chacun des sommets du graphe.



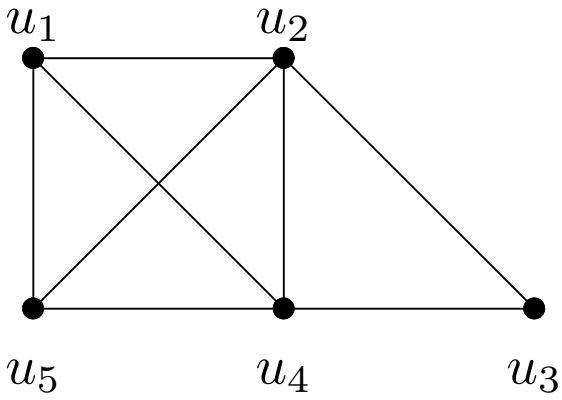
Sommet	Sommets adjacents
a	b, d
b	a, c, d, e
c	d, e
d	a, b, c
e	b, c

Définition: matrice d'adjacence

Soit $G = (V, E)$ un graphe simple, tq $|V| = n$.
Supposons que les sommets de G sont, arbitrairement, $v_1, v_2, ..., v_n$.

La **matrice d'adjacence A** (ou matrice associée $\mathbf{A}_G$) de G se rapportant à cet ensemble de sommets est la matrice booléenne $M = (n, n)$ telle que :
- $a_{ij} = 1$ quand v_i et v_j sont adjacents et,
- $a_{ij} = 0$ sinon.

Exemple de matrice d'adjacence



Les sommets sont dans l'ordre u_1, u_2, u_3, u_4, u_5.

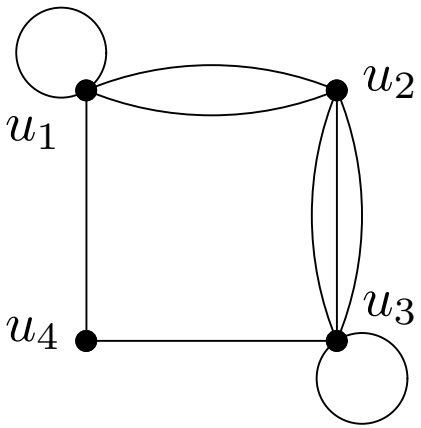
$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Remarque sur les matrices d'adjacence

- L'ordre avec lequel on numérote les n sommets du graphe est arbitraire. Il y a $n!$ façons d'ordonner n sommets et donc un graphe donné peut être représenté par $n!$ matrices d'adjacence différentes.
- La matrice d'adjacence d'un graphe simple est symétrique car si v_i est adjacent à v_j, alors v_j est adjacent à v_i et de même si v_i n'est pas adjacent à v_j, alors v_j n'est pas adjacent à v_i.
- Puisqu'un graphe simple n'a pas de boucle, $a_{ii} = 0$ pour $i = 1, 2, ..., n$.

Matrice d'adjacence pour les pseudographes

- Une boucle sur le sommet v_i est notée par un 1 à la (i, i)-ième position de la matrice d'adjacence.
- Quand il y a des arcs multiples, le (i, j)-ième élément de la matrice d'adjacence est égal au nombre d'arcs qui sont associés à $\{v_i, v_j\}$.
- Tous les graphes non orientés, incluant les multigraphes et les pseudographes, ont des matrices d'adjacence symétriques.



$$\begin{pmatrix} 1 & 2 & 0 & 1 \\ 2 & 0 & 3 & 0 \\ 0 & 3 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

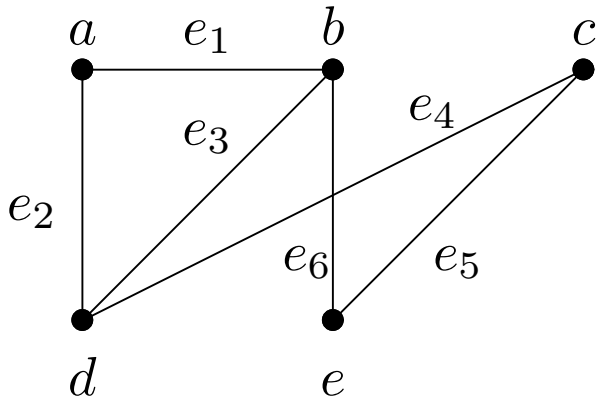
Définition: matrice d'incidence

Soit $G = (V, E)$ est un graphe simple, où $|V| = n$ et $|E| = m$. On suppose aussi que les sommets de G sont, arbitrairement, $v_1, v_2, ..., v_n$ et que les arcs de G sont, arbitrairement, $e_1, e_2, ..., e_m$.

La **matrice d'incidence** de G correspondant à cet ordonnancement de V et E est la matrice $\mathbf{M} = [m_{ij}]$ de dimension $n \times m$ où

$$m_{ij} = \begin{cases} 1 & \text{quand l'arc } e_j \text{ est incident au sommet } v_i, \\ 0 & \text{autrement.} \end{cases}$$

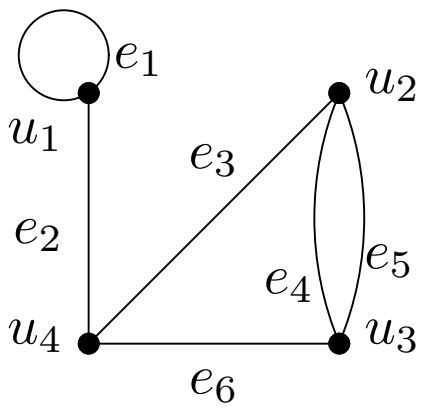
Exemple de matrice d'incidence



	e_1	e_2	e_3	e_4	e_5	e_6
a	1	1	0	0	0	0
b	1	0	1	0	0	1
c	0	0	0	1	1	0
d	0	1	1	1	0	0
e	0	0	0	0	1	1

Matrice d'incidence pour les pseudographes

- Les boucles sont représentées en utilisant une colonne qui comprend exactement une entrée égale à 1, ce qui correspond au sommet qui est incident à cette boucle.
- Les arcs multiples sont représentés dans la matrice d'incidence en remplissant les colonnes avec des valeurs identiques puisque ces arcs sont incidents à la même paire de sommets.



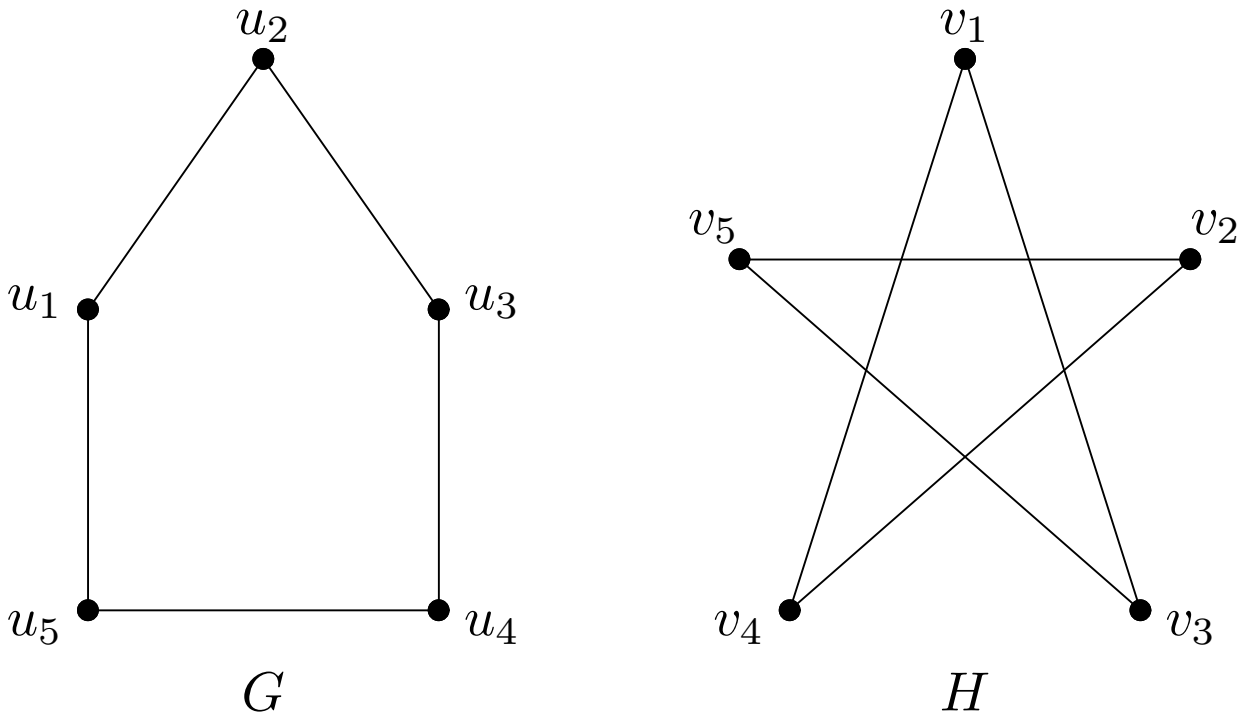
	e_1	e_2	e_3	e_4	e_5	e_6
u_1	1	1	0	0	0	0
u_2	0	0	1	1	1	0
u_3	0	0	0	1	1	1
u_4	0	1	1	0	0	1

Définition: isomorphisme de graphes

Les graphes simples $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$ sont **isomorphes** s'il existe une fonction bijective f de V_1 dans V_2 avec la propriété suivante: a et b sont adjacent dans G_1 si et seulement si $f(a)$ et $f(b)$ sont adjacent dans G_2 pour toutes les valeurs de a et de b dans V_1.

Lorsque deux graphes simples sont isomorphes, il existe une bijection entre les sommets de ces deux graphes qui préserve la relation d'adjacence.

Exemple de graphes isomorphes



$f(u_1) = v_1$, $f(u_2) = v_3$, $f(u_3) = v_5$, $f(u_4) = v_2$ et $f(u_5) = v_4$.

Exemple de graphes isomorphes

G	u_1	u_2	u_3	u_4	u_5
u_1	0	1	0	0	1
u_2	1	0	1	0	0
u_3	0	1	0	1	0
u_4	0	0	1	0	1
u_5	1	0	0	1	0

H	v_1	v_2	v_3	v_4	v_5
v_1	0	0	1	1	0
v_2	0	0	0	1	1
v_3	1	0	0	0	1
v_4	1	1	0	0	0
v_5	0	1	1	0	0

G	u_1	u_2	u_3	u_4	u_5
u_1	0	1	0	0	1
u_2	1	0	1	0	0
u_3	0	1	0	1	0
u_4	0	0	1	0	1
u_5	1	0	0	1	0

H	v_1	v_3	v_5	v_2	v_4
v_1	0	1	0	0	1
v_3	1	0	1	0	0
v_5	0	1	0	1	0
v_2	0	0	1	0	1
v_4	1	0	0	1	0

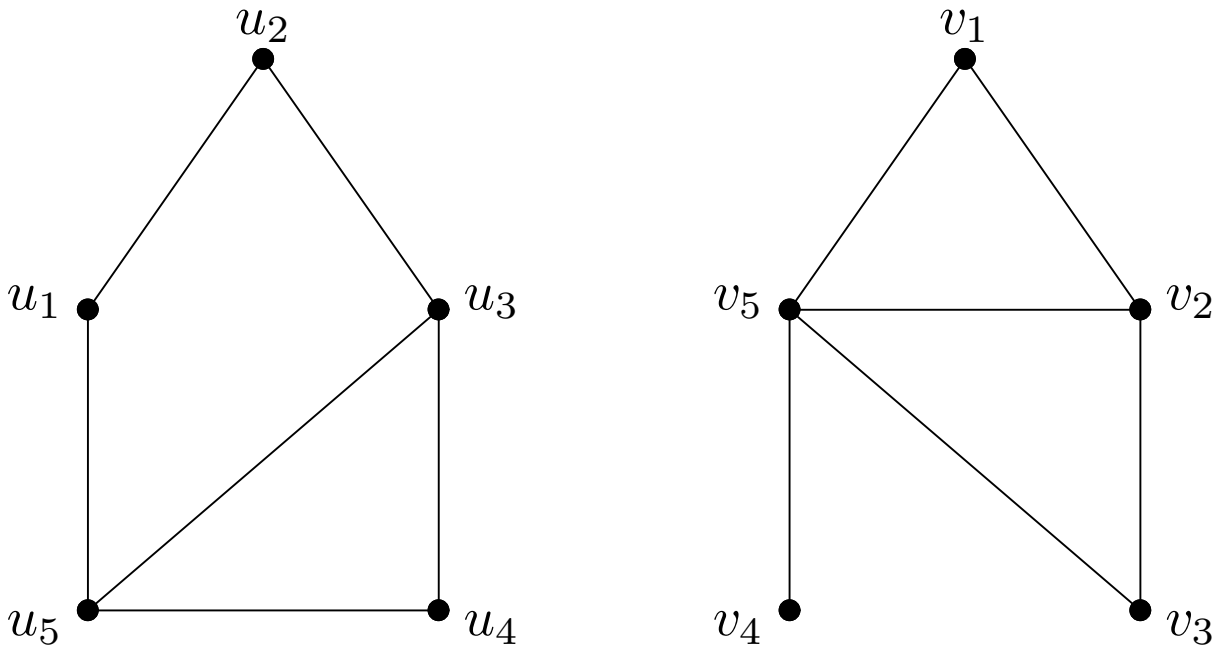
Invariants par rapport à l'isomorphisme

Il y a des propriétés qui sont invariantes entre deux graphes isomorphes. Par exemple, deux graphes isomorphes simples doivent

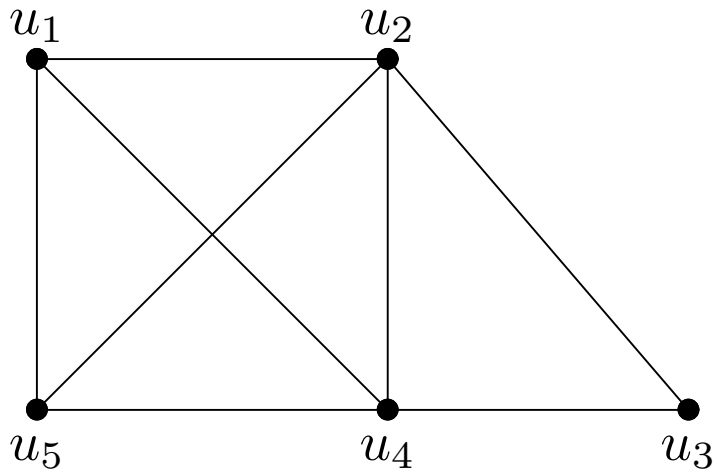
- avoir le même nombre de sommets,
- avoir le même nombre d'arcs,
- les mêmes degrés pour les sommets.

Note 1: Ces conditions sont nécessaires mais pas suffisantes pour montrer que deux graphes sont isomorphes.
Note 2: Le non respect d'une de ces condition est suffisant mais non nécessaire pour montrer que deux graphes ne sont pas isomorphes.

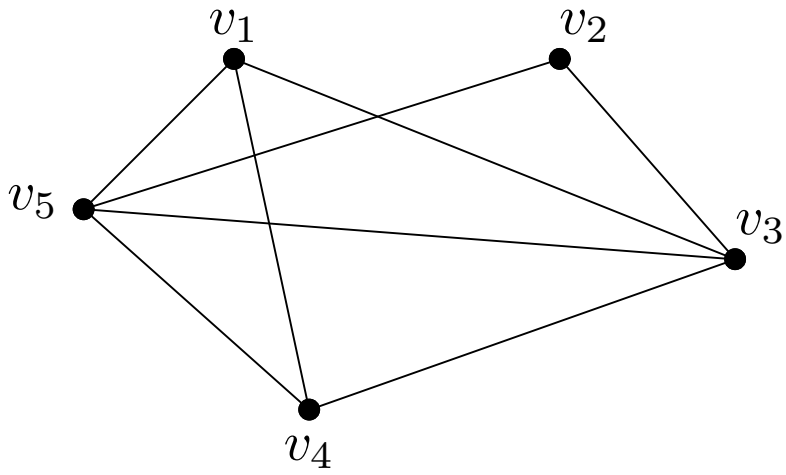
Exemple de graphes non isomorphes



Ces deux graphes sont-ils isomorphes?

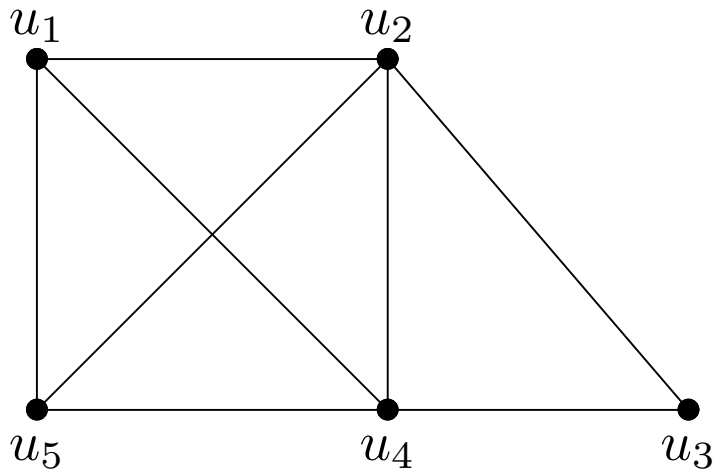


G

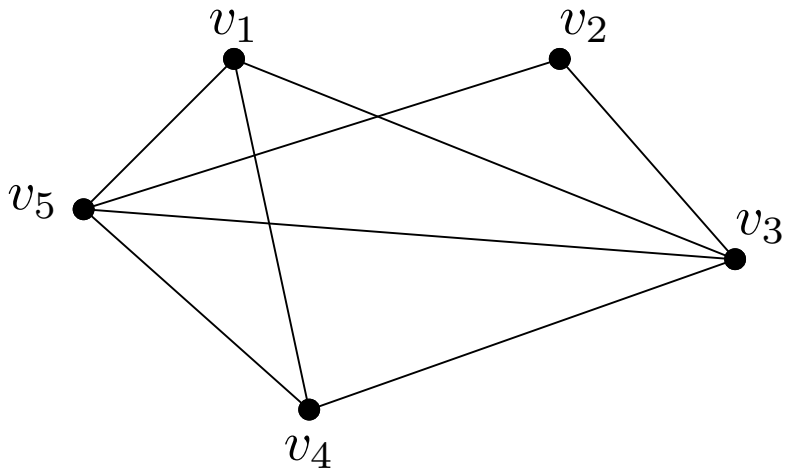


H

Ces deux graphes sont-ils isomorphes?



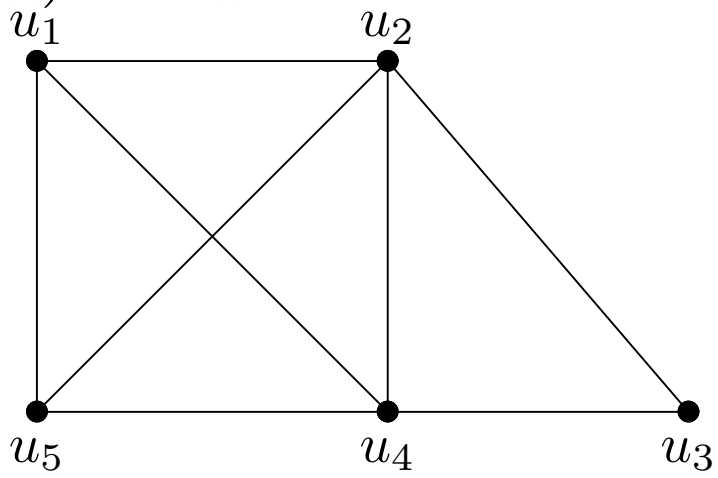
G



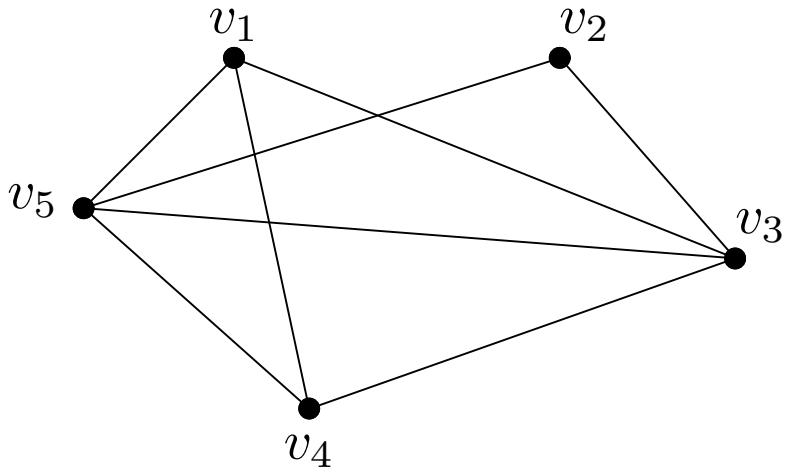
H

Ces deux graphes sont isomorphes

$f(u_3) = v_2$, $f(u_4) = v_3$, $f(u_2) = v_5$, $f(u_5) = v_4$ et $f(u_1) = v_1$.



G



H

G	u_1	u_2	u_3	u_4	u_5
u_1	0	1	0	1	1
u_2	1	0	1	1	1
u_3	0	1	0	1	0
u_4	1	1	1	0	1
u_5	1	1	0	1	0

H	v_1	v_5	v_2	v_3	v_4
v_1	0	1	0	1	1
v_5	1	0	1	1	1
v_2	0	1	0	1	0
v_3	1	1	1	0	1
v_4	1	1	0	1	0

Module Parcours de graphes

Parcours de graphes

Chapitre .1 Connexité, chaînes et chemins (64)

- Chaîne (64) • Cycle (65)
- Chemin (67) • Circuit (68)
- Connexité (70) • Composantes connexes (71)
- Points de coupure (73) • Séparateur (75)
- Invariants par rapport à l'isomorphisme (79)
- Dénombrement des chemins (80)

Suite page suivante...

Module

...suite de la page précédente

Chapitre .2 Cycles eulériens et hamiltoniens (84)

- Problème de Königsberg (84)
- Chaîne et cycle eulériens (87)
- Algorithme de construction d'un cycle eulérien (92)
- Chaîne et cycle hamiltoniens (102)
- Code Gray (104)

Chapitre .3 Graphes valués et chemins minimaux (108)

- Longueur du chemin (109)
- Algorithme du chemin minimal de Dijkstra (112)
- Algorithme du chemin minimal de Floyd (123)

Définition: chaîne

Une **chaîne** de longueur n de u à v, où n est un entier positif dans un graphe non orienté, est une séquence d'arcs $e_1, e_2, ..., e_n$ du graphe, de telle sorte que $f(e_1) = \{x_0, x_1\}$, $f(e_2) = \{x_1, x_2\}$, ..., $f(e_n) = \{x_{n-1}, x_n\}$, où $x_0 = u$ et $x_n = v$.

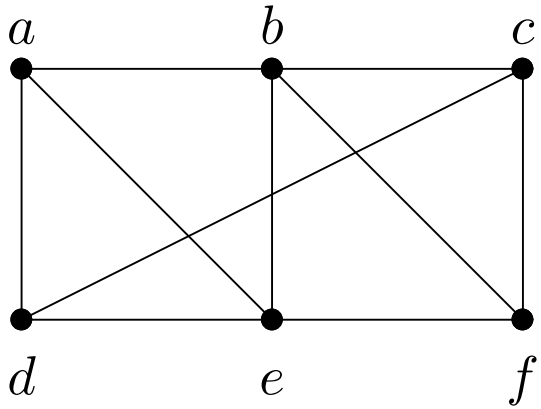
Quand le graphe est simple, on définit cette chaîne au moyen de la séquence des sommets parcourus $x_0, x_1, ..., x_n$, puisque la liste de ces sommets détermine la chaîne de manière unique.

Définition: cycle, cycle simple

Dans un graphe non orienté, une chaîne est appelée un **cycle** si elle commence et se termine au même sommet, autrement dit si $x_0 = u = v = x_n$.

Dans un graphe non orienté, une chaîne ou un cycle est **simple** s'il ne passe pas par le même arc plus d'une fois.

Exemples de chaînes et de cycles



a, d, c, f, e est une chaîne simple de longueur 4.

d, e, c, a n'est pas une chaîne parce que $\{e, c\}$ et $\{c, a\}$ ne sont pas des arcs.

b, c, f, e, b est un cycle simple de longueur 4.

a, b, e, d, a, b de longueur 5 n'est pas une chaîne simple puisqu'elle comprend l'arc $\{a, b\}$ deux fois.

Définition: chemin

Un **chemin** de longueur n de u à v, où n est un entier positif dans un multigraphe orienté, est une séquence d'arcs $e_1, e_2, ..., e_n$ du graphe, de telle sorte que $f(e_1) = (x_0, x_1)$, $f(e_2) = (x_1, x_2)$, ..., $f(e_n) = (x_{n-1}, x_n)$, où $x_0 = u$ et $x_n = v$.

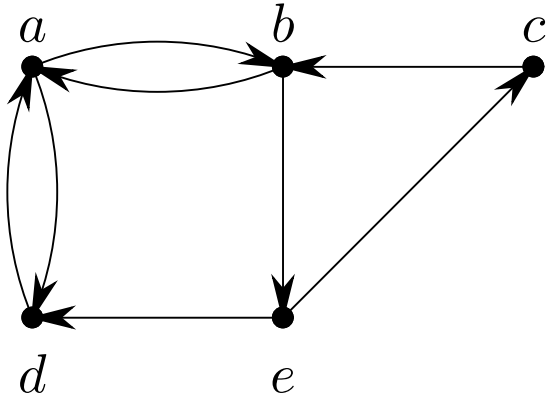
Quand le graphe ne comprend pas d'arcs multiples, on définit ce chemin au moyen de la séquence des sommets parcourus $x_0, x_1, ..., x_n$, puisque la liste de ces sommets détermine le chemin de manière unique.

Définition: circuit, circuit simple

Dans un multigraphe orienté, un chemin est appelé un **circuit** si il commence et se termine au même sommet, autrement dit si $x_0 = u = v = x_n$.

Dans un multigraphe orienté, un chemin ou un circuit est **simple** s'il ne passe pas par le même arc plus d'une fois.

Exemples de chemins et de circuits



a, b, e, c, b est un chemin simple de longueur 4.

a, d, a, d, a est un circuit de longueur 4.

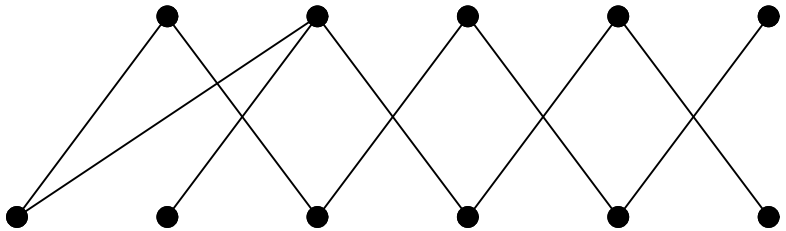
a, d, b, e, a n'est pas un chemin par ce que (d, b) n'est pas un arc.

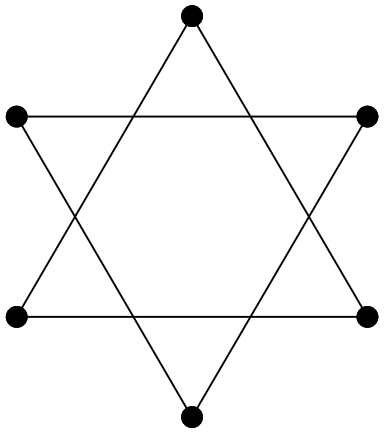
a, b, e, c, b, d, a n'est pas un chemin par ce que (b, d) n'est pas un arc.

Définition: connexité

Un graphe non orienté est **connexe** s'il existe une chaîne entre n'importe quelle paire de sommets distincts du graphe.

Graphe connexe:

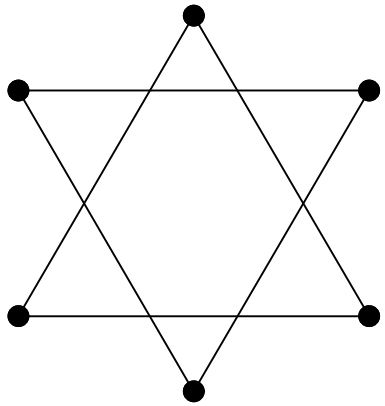




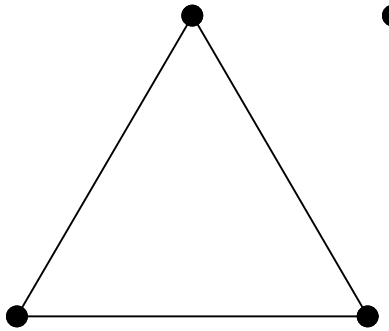
Graphe non connexe:

Définition: composantes connexes

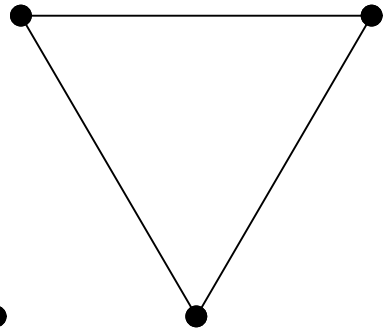
Un graphe qui n'est pas connexe est l'union de deux ou de plusieurs sous-graphes connexes, chaque paire de ceux-ci n'ayant pas de sommet en commun. Les sous-graphes disjoints sont les **composantes connexes** du graphe.



G



G_1



G_2

Chaîne simple dans un graphe connexe

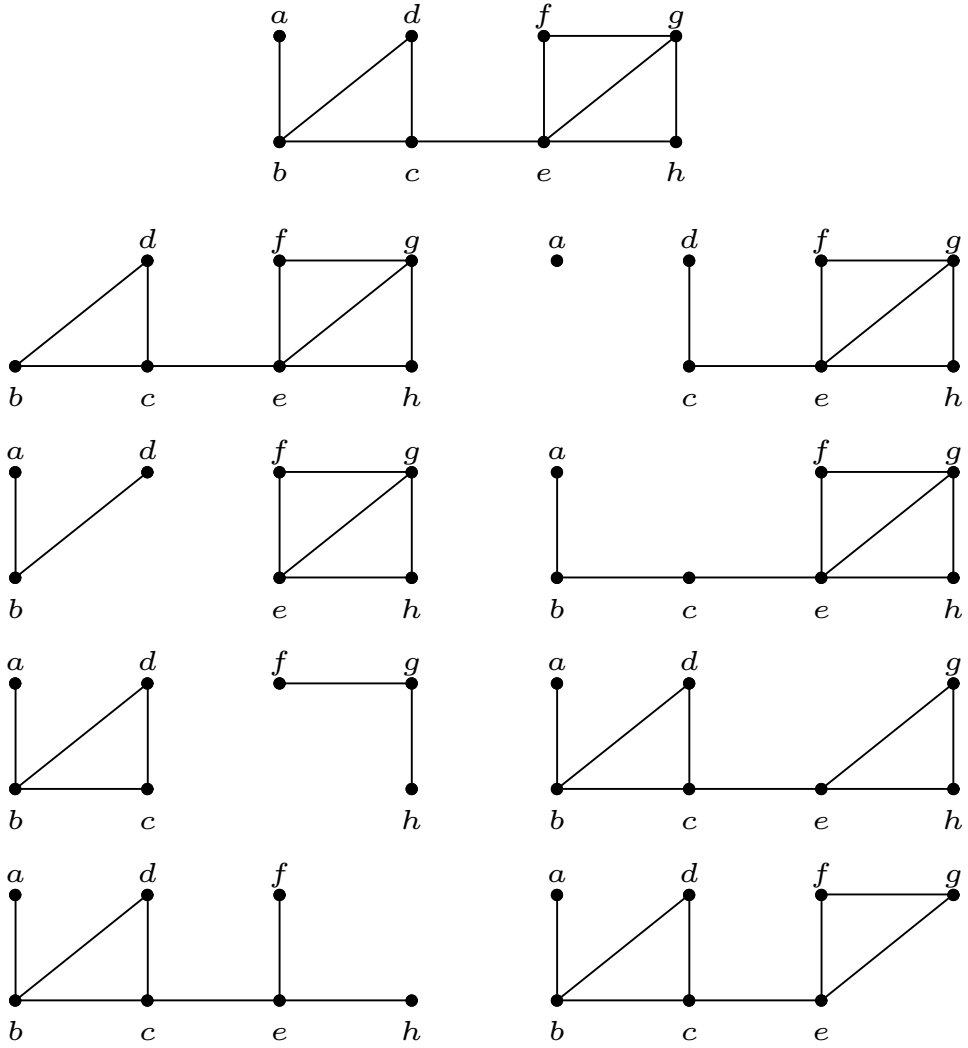
THÉORÈME: Il existe une chaîne simple entre n'importe quelle paire de sommets distincts d'un graphe non orienté connexe.

PREUVE : Soient u et v 2 sommets distincts de $G = (V, E)$. G est connexe, il existe une chaîne entre u et v. Soient $x_0 = u, x_1, ..., x_n = v$ la chaine de longueur minimale. Cette chaîne est simple. On le prouve par l'absurde: Supposons qu'il existe i et j tq $x_i = x_j$ et $o \leq i \leq j$. Cela signifie qu'il existe une chaine $x_0, x_1, ..., x_{i-1}, x_j, ..., x_n$ entre u et v de longueur plus courte, obtenue en supprimamt les arcs correspondant aux sommets $x_i, ..., x_{j-1}$.

Définition: points de coupure

Un sommet est appelé un **point de coupure** ou **point d'articulation** si le retrait de ce sommet et de tous les arcs incidents à ce sommet conduit à former un sous-graphe ayant plus de composantes connexes que le graphe initial.

Exemples de points de coupure

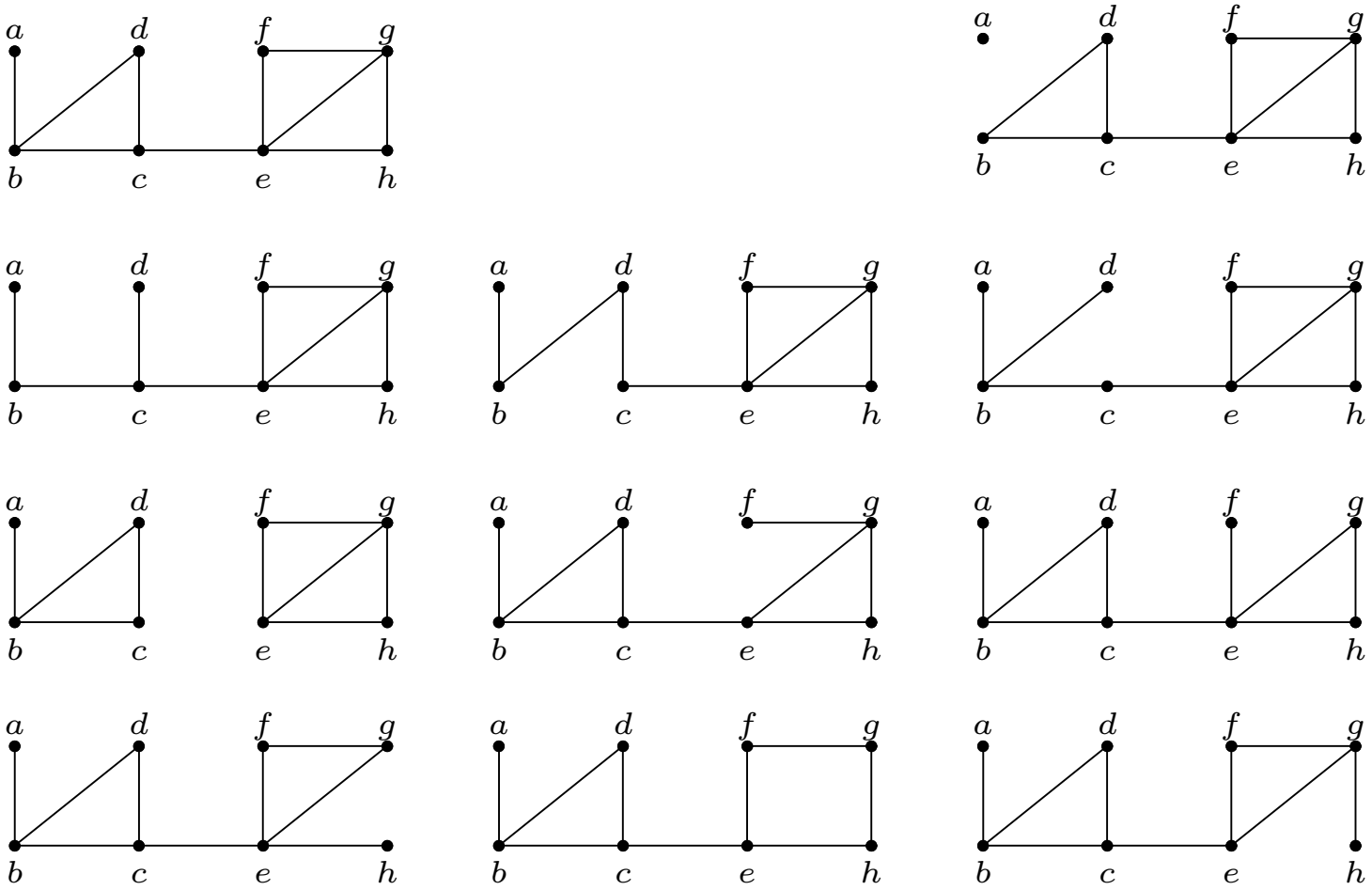


Les sommets b, c et e sont des points de coupure.

Définition: séparateur

Un arc est appelé un **séparateur** ou un **pont** si le retrait de cet arc conduit à former un sous-graphe ayant plus de composantes connexes que le graphe initial.

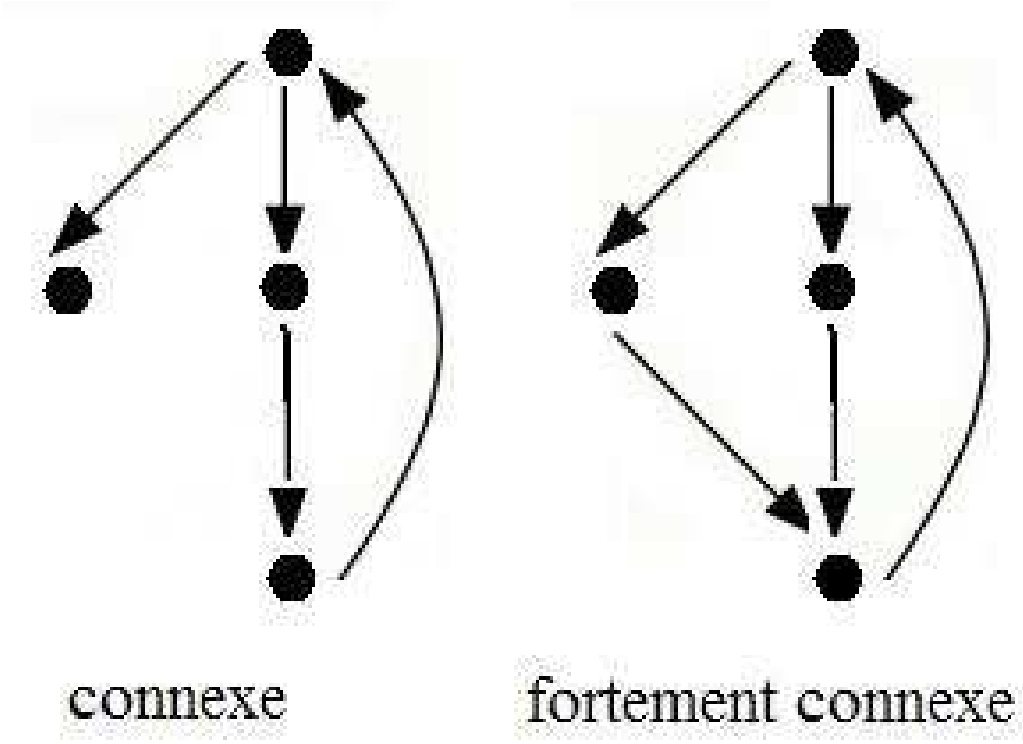
Exemples d'arcs séparateurs



Les arcs $\{a, b\}$ et $\{c, e\}$ sont des séparateurs.

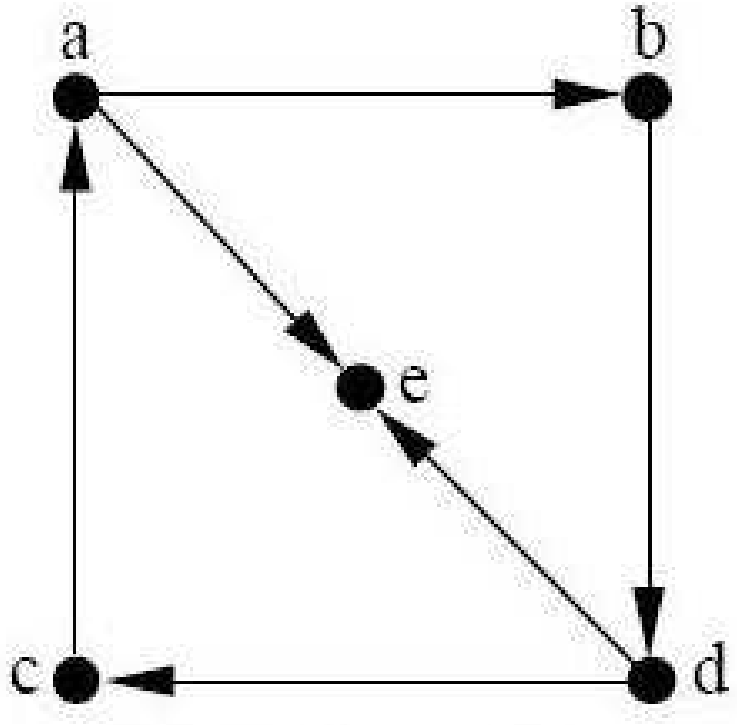
Définition: graphe orienté fortement connexe

Un graphe orienté est **fortement connexe** si, pour tout couple de sommets (a, b) du graphe, il existe un chemin de a à b et de b à a.



Définition: graphe orienté faiblement connexe

Un graphe orienté est **faiblement connexe** s'il existe une chaîne entre n'importe quelle paire de sommets dans le graphe non orienté sous-jacent.

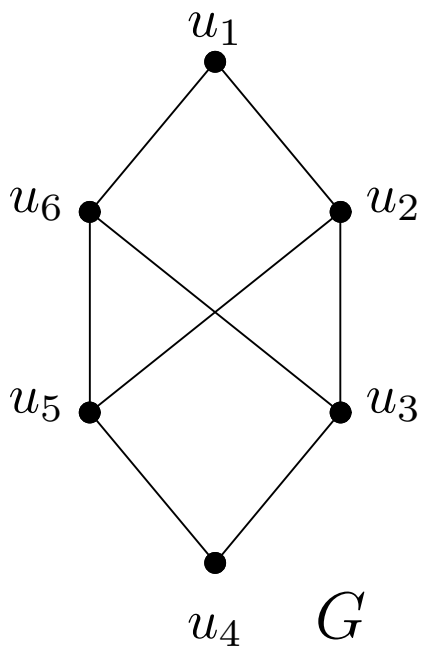


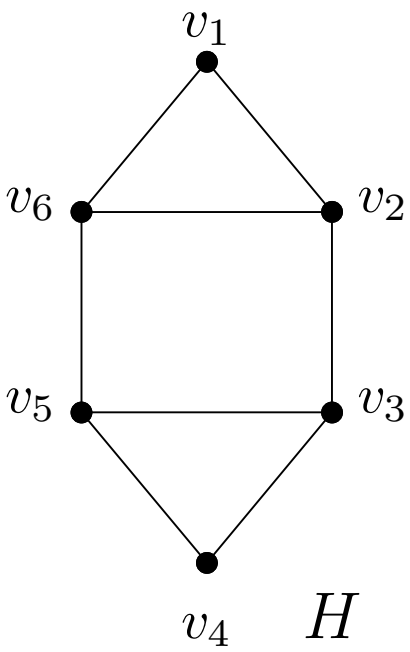
faiblement connexe

Un graphe fortement connexe est faiblement connexe.

Invariants par rapport à l'isomorphisme

L'existence d'un cycle simple de longueur k, où k est un nombre entier plus grand que 2, est une autre propriété invariante entre deux graphes isomorphes. Cette condition est nécessaire mais pas suffisante pour montrer que deux graphes sont isomorphes.





Tip : G n'a pas de circuit de longueur 3.

Dénombrement des chemins

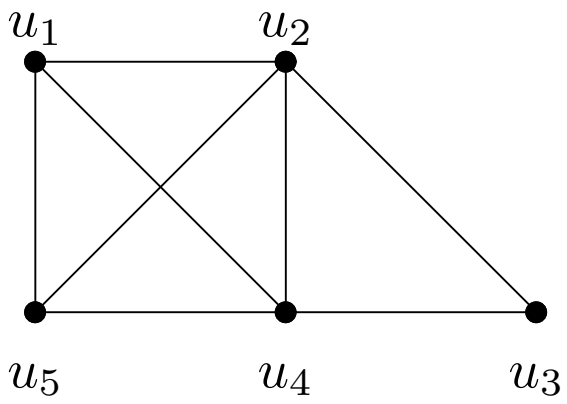
THÉORÈME: Soit G un graphe, avec des arcs orientés ou non orientés, avec la possibilité d'arcs multiples et de boucles, et ayant une matrice d'adjacence $\mathbf{A}$ en présumant l'ordonnancement $v_1, v_2, ..., v_n$ des sommets de G.

Le nombre de chemins différents de longueur p de v_i à v_j, où p est un nombre entier, est égale au (i, j)-ième élément de $\mathbf{A}^p$.

Note: Il s'agit ici de la puissance standard de $\mathbf{A}$, pas du produit booléen.

PREUVE par induction sur la longueur des chemins.

Exemple de dénombrement

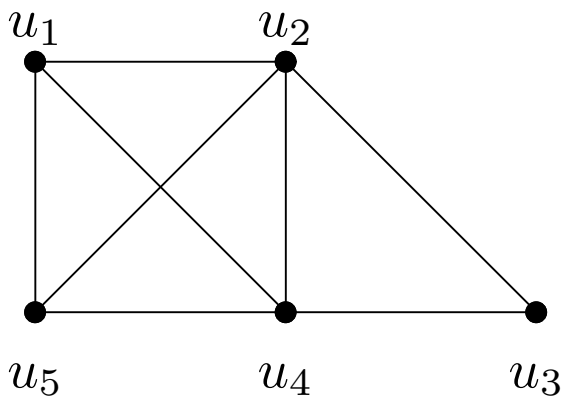


Les sommets sont dans l'ordre u_1, u_2, u_3, u_4, u_5.

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

Nombre de chemins de longueur 1.

Exemple de dénombrement

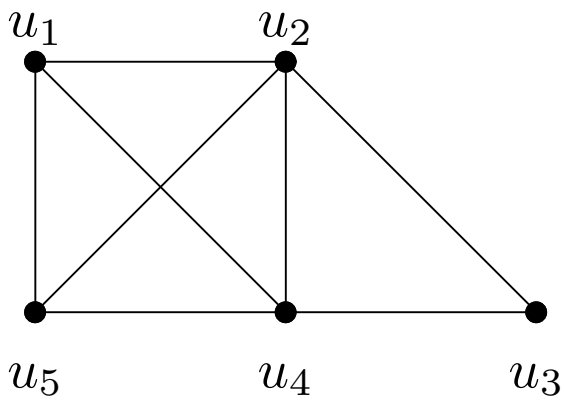


Les sommets sont dans l'ordre u_1, u_2, u_3, u_4, u_5.

$$\mathbf{A}^2 = \begin{pmatrix} 3 & 2 & 2 & 2 & 2 \\ 2 & 4 & 1 & 3 & 2 \\ 2 & 1 & 2 & 1 & 2 \\ 2 & 3 & 1 & 4 & 2 \\ 2 & 2 & 2 & 2 & 3 \end{pmatrix}$$

Nombre de chemins de longueur 2.

Exemple de dénombrement

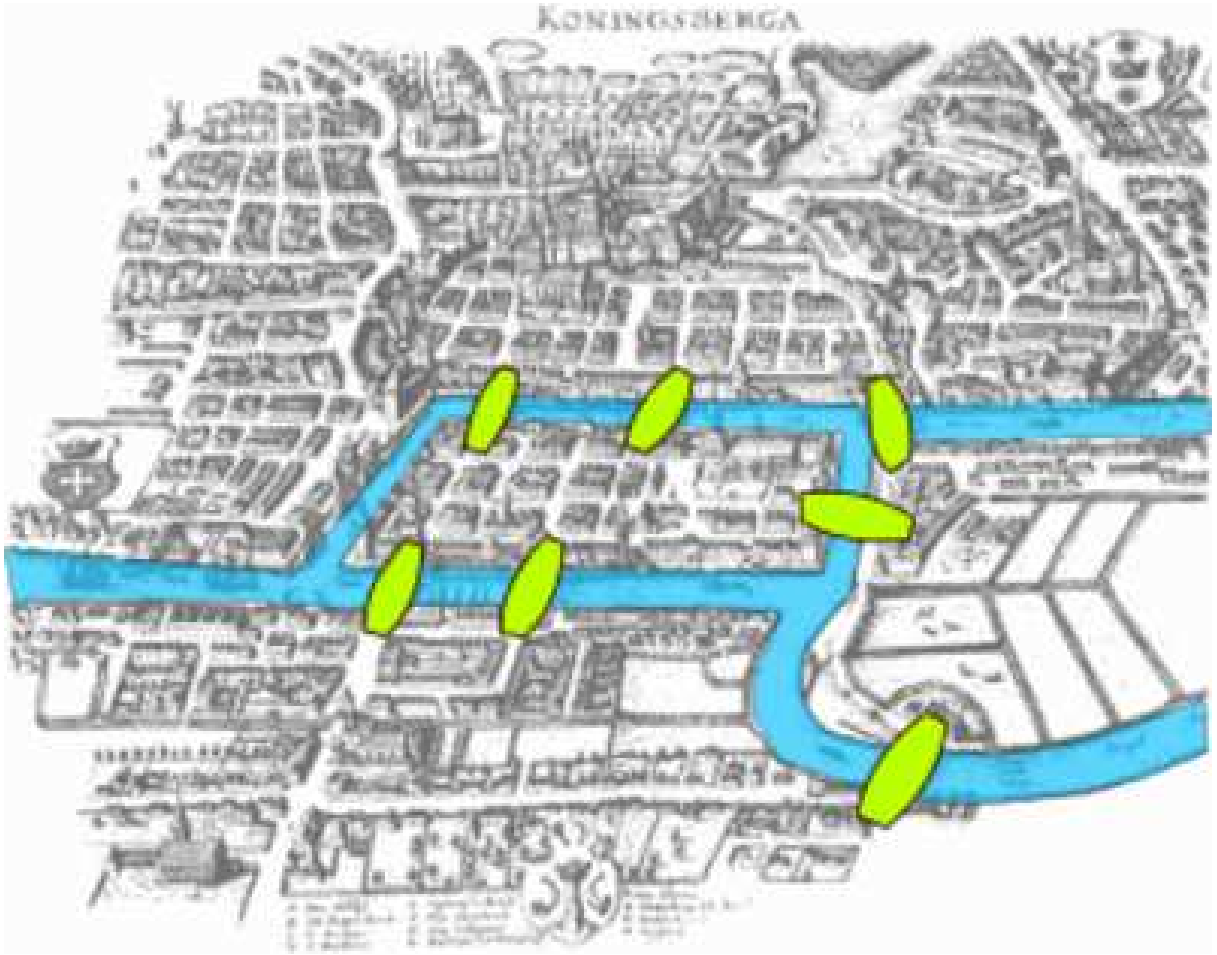


Les sommets sont dans l'ordre u_1, u_2, u_3, u_4, u_5.

$$\mathbf{A}^3 = \begin{pmatrix} 6 & 9 & 4 & 9 & 7 \\ 9 & 8 & 7 & 9 & 9 \\ 4 & 7 & 2 & 7 & 4 \\ 9 & 9 & 7 & 8 & 9 \\ 7 & 9 & 4 & 9 & 6 \end{pmatrix}$$

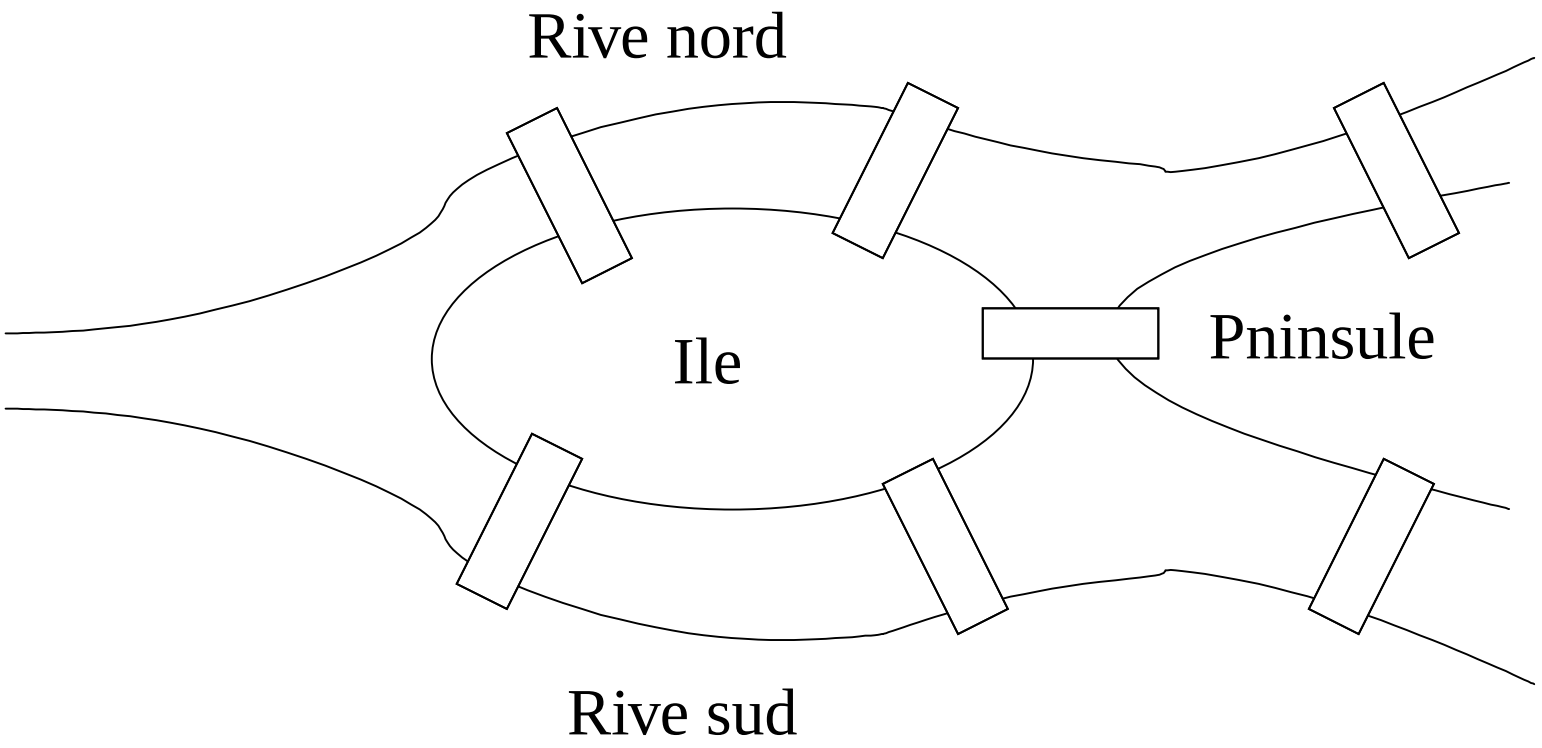
Nombre de chemins de longueur 3.

Problème de Königsberg



fr.wikipedia.org/wiki/Sept_ponts_de_K%C3%B6nigsberg

Problème de Königsberg



Problème: Partir et revenir au même endroit en traversant une et une seule fois chacun des sept ponts.

Note historique: Leonhard Euler

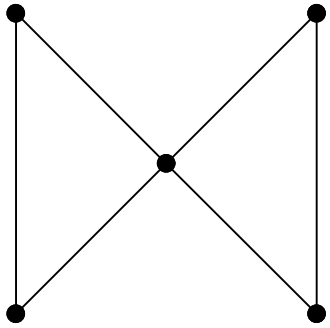
Né le 15 avril 1707 à Bâle, Suisse.

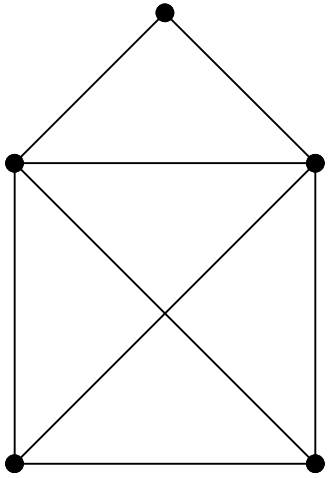
Mort le 18 septembre 1783 à St-Petersbourg, Russie.

`www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Euler.html`

Définition: chaîne et cycle eulériens

Une **chaîne eulérienne** dans un graphe G est une chaîne simple qui contient tous les arcs de G.
Un **cycle eulérien** dans un graphe G est un cycle simple contenant tous les arcs de G.





Condition d'existence d'un cycle eulérien

DÉFINITION (Graphe connexe eulérien). Un graphe connexe est eulérien s'il existe un cycle eulérien.

THÉORÈME: Un multigraphe connexe est eulérien si et seulement si chacun de ses sommets est de degré pair.

Preuve pages suivantes.

Démonstration : Lemmes

DÉFINITION 1 : On appelle cheminement maximal d'origine x toute chaîne de la forme $\mu = (x, ..., y)$ telle que toutes les arêtes incidentes à y sont dans μ, i.e. on ne peut plus prolonger la chaîne à partir de y.

LEMME 1 : Soit $G = (X, E)$ un graphe connexe, dont tous les sommets ont un degré pair, alors tout cheminement maximal est un cycle.

LEMME 2 : On considère un graphe connexe $G = (X, E)$ et un cycle ν de ce graphe, alors toutes les composantes connexes $C_1, C_2, ..., C_p$ de $G \backslash \nu$ (on retire les arêtes, pas les sommets) ont chacune un sommet en commun avec ν.

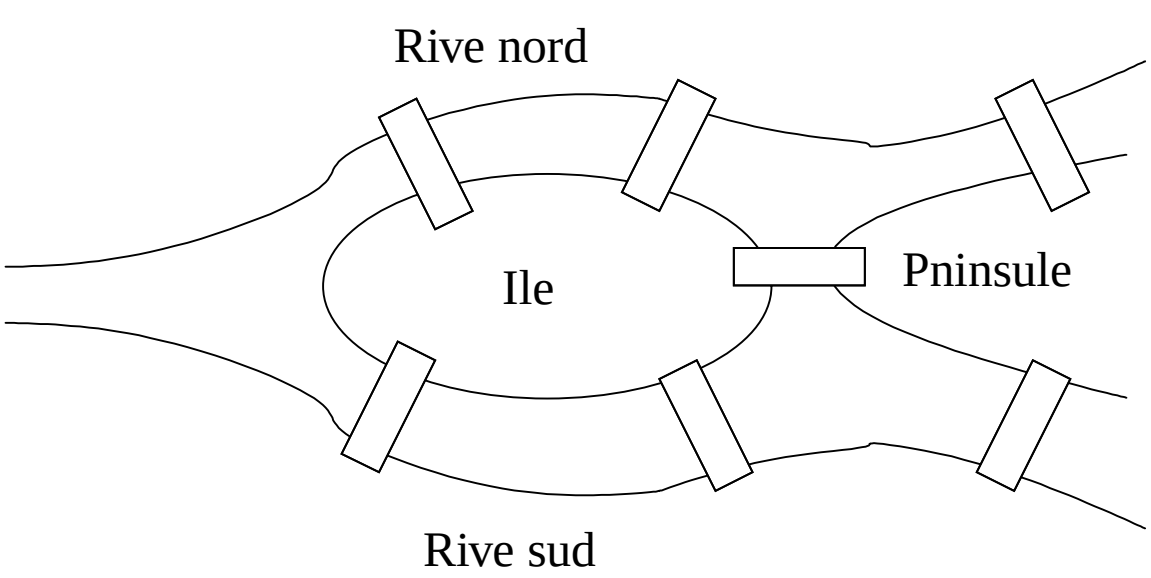
Démonstrations en exercice

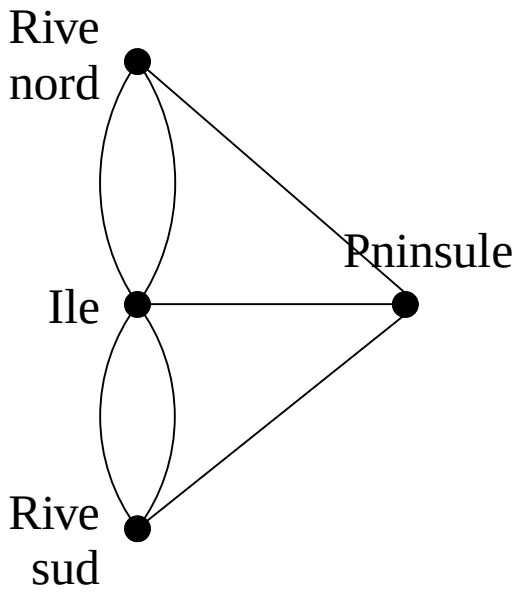
Démonstration du théorème

$\Rightarrow$ Soit G un graphe connexe, eulérien. Alors, il existe un cycle $\mu = (x_1, x_2, ..., x_n, x_1)$ passant une fois et une seule par toutes les arêtes. μ peut être considéré comme un cheminement maximal, donc $d(x1)$ est pair (Lemme 1). De même, en considérant le cheminement maximal $(x_2, ..., x_n, x_1, x_2)$, on trouve que $d(x2)$ est pair. En itérant le procédé, on trouve que $\forall x \in X, d(x) pair$.

$\Leftarrow$ Soient ν un cheminement maximal (donc un cycle), C une composante connexe de $G \backslash \nu$. Pour tout sommet x, les arêtes de ν adjacentes à x sont en nombre pair. Donc les sommets de C sont de degré pair. On obient récursivement un cycle eulérien μ dans C. Soit x un sommet commun à ν et μ (existant grâce au lemme 2). On peut donc écrire $\mu = (\mu_1, ..., \mu_{i-1}, x, \mu_{i+1}, ..., \mu_{|\mu|})$ et $\nu = (\nu_1, ..., \nu_{i-1}, x, \nu_{i+1}, ..., \nu_{|\nu|})$. On combine alors ces deux cycles en un seul cycle $\mu = (\mu_1, ..., \mu_{i-1}, x, \nu_{j+1}, ..., \nu_{|\mu|}, \nu_1, ..., \nu_{j-1}, x, \mu_{i+1}, ..., \mu_{|\mu|})$. On fait de même avec les autres composantes connexes, et l'on obtient un cycle eulérien.

Problème de Königsberg





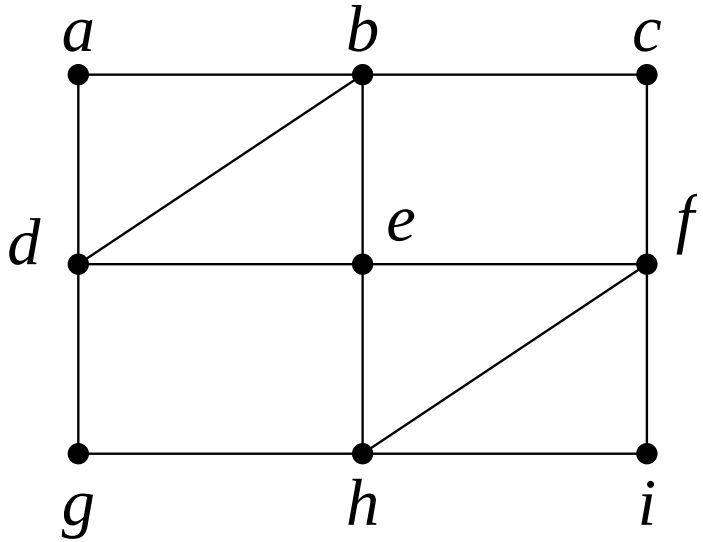
Problème: Partir et revenir au même endroit en traversant une et une seule fois chacun des sept ponts. Ce problème n'a pas de solution.

Algorithme de construction d'un cycle eulérien

procédure *Euler*(G: multigraphe connexe avec tous les
sommets de degré pair)
cycle := Un cycle dans G commençant à un sommet arbitraire
comprenant des arc ajoutés de manière à former une
chaîne qui retourne à ce premier sommet.
H := G moins les arcs du cycle.
tant que H a au moins un arc
début
sous-cycle := un cycle dans H qui commence à un
sommet de H qui est un sommet du *cycle*.
H := H moins les arcs du *sous-cycle*.
cycle := *cycle* avec *sous-cycle* inséré au sommet
approprié.
fin {*cycle* est un cycle eulérien}

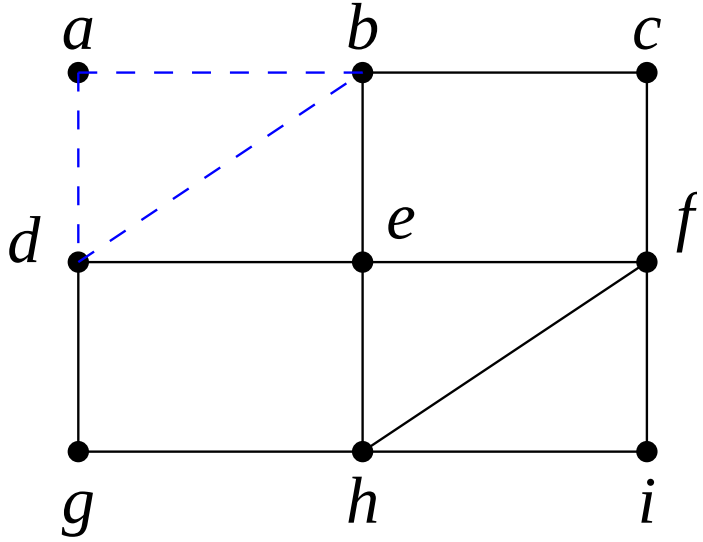
Exemple de construction d'un cycle eulérien

Problème initial:



Exemple de construction d'un cycle eulérien

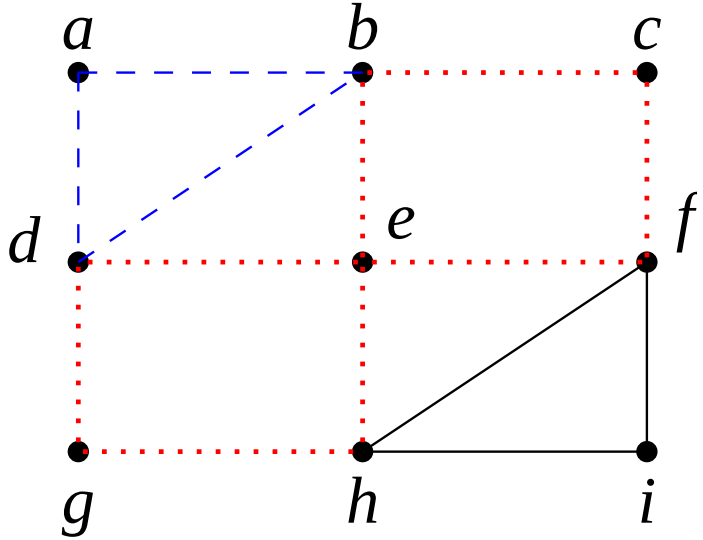
Étape 1 de 3:



cycle := a, d, b, a

Exemple de construction d'un cycle eulérien

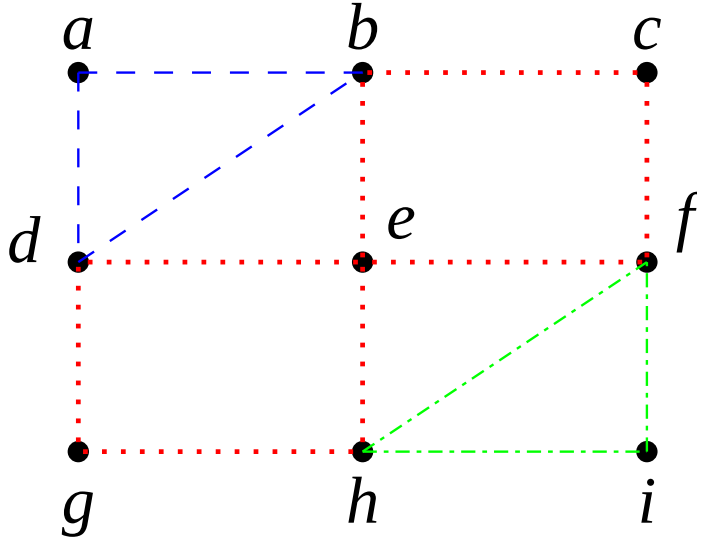
Étape 2 de 3:



cycle := a, d, b, a
sous-cycle := $d, e, f, c, b, e, h, g, d$
cycle := $a, d, e, f, c, b, e, h, g, d, b, a$

Exemple de construction d'un cycle eulérien

Étape 3 de 3:



cycle := $a, d, e, f, c, b, e, h, g, d, b, a$
sous-cycle := f, i, h, f
cycle := $a, d, e, f, i, h, f, c, b, e, h, g, d, b, a$

Condition d'existence d'une chaîne eulérienne

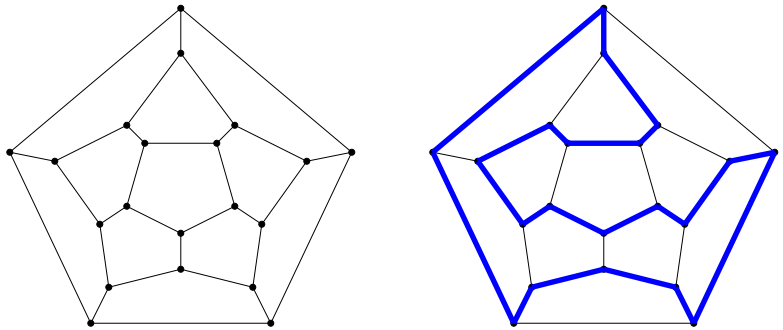
THÉORÈME: Un multigraphe connexe admet une chaîne eulérienne et non un cycle eulérien, si et seulement si, il a exactement deux sommets de degré impair.

PREUVE :

Supposons que le graphe connexe a une chaîne eulérienne entre a et b mais pas de cycle eulérien. Le premier arc de la chaîne contribue pour 1 au degré de a. Chaque nouveau passage par a contribue pour 2 à son degré. Le dernier arc de la chaîne contribue pour 1 au degré de b et chaque nouveau passage par b contribue pour 2 à son degré. Par conséquent a et b ont un degré impair et les autres sommets ont un degré pair puisque la chaîne contribue pour 2 à leur degré.

Inversement, supposons qu'un graphe G a exactement 2 sommets de degré impair, disons a et b. Considérons le graphe composé de G plus l'arc $\{a, b\}$. Chaque sommet du nouveau graphe a un degré pair et par conséquent il contient un cycle eulérien. Si on enlève l'arc a, b on obtient une chaîne eulérienne.

Note historique: Sir William Rowan Hamilton



Né le 4 août 1805 à Dublin, Irlande.

Mort le 2 septembre 1865 à Dublin, Irlande.

Mathématicien et astronome, inventeur des quaternions.

`www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Hamilton.html`

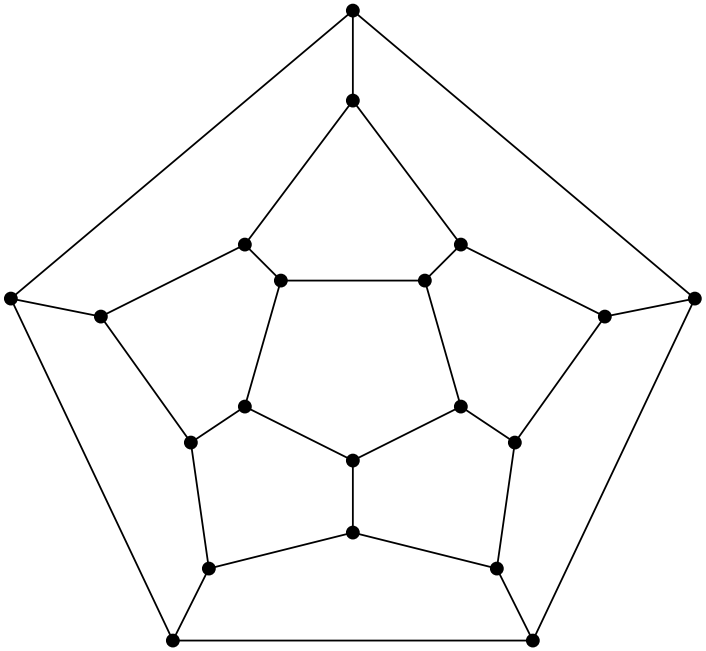
Casse-tête du “Tour du monde”

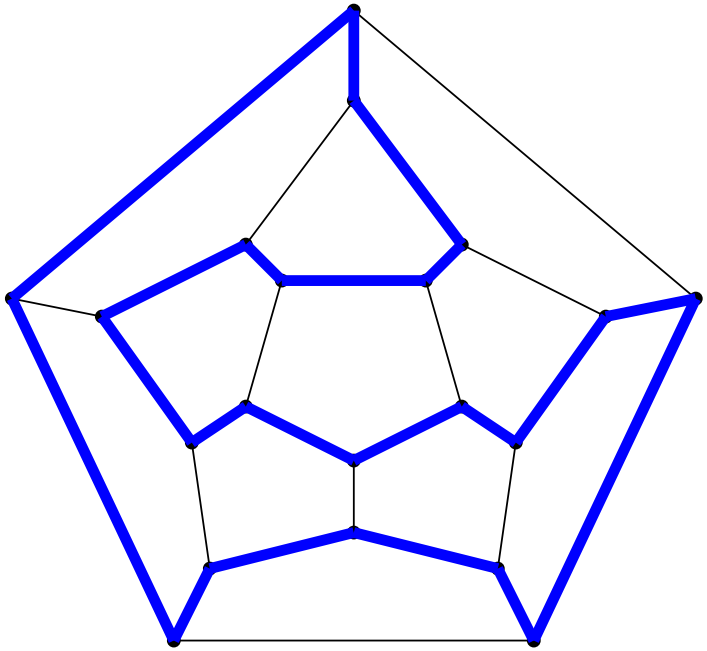
www.puzzlemuseum.com/month/picm02/200207icosian.htm

Casse-tête du “Tour du monde”

www.puzzlemuseum.com/month/picm02/200207icosian.htm

Solution du “Tour du monde” d’Hamilton





Définitions: chaîne et cycle hamiltoniens

Une chaîne $x_0, x_1, ..., x_{n-1}, x_n$ dans le graphe $G = (V, E)$ est appelée une **chaîne hamiltonienne** si $V = \{x_0, x_1, ..., x_{n-1}, x_n\}$ et $x_i \neq x_j$ pour $0 \leq i < j \leq n$.

Un cycle $x_0, x_1, ..., x_{n-1}, x_n, x_0$ dans le graphe $G = (V, E)$ est appelé un **cycle hamiltonien** si $x_0, x_1, ..., x_n$ est une chaîne hamiltonienne.

Un graphe est hamiltonien s'il est possible de trouver un cycle passant une et une seule fois par tous les sommets.

Conditions d'existence des cycles hamiltoniens

Condition nécessaire :
Un graphe avec un sommet de degré 1 ne peut contenir de cycle hamiltonien.

Si un sommet est de degré 2, alors les deux arcs incidents à ce sommet doivent faire partie d'un cycle hamiltonien.

Condition suffisante:
Si G est un graphe simple connexe avec n sommets où $n \geq 3$, alors G à un cycle hamiltonien si le degré de chaque sommet est au moins égal à $n/2$.

Note historique: Émile Baudot

Les codes Gray (non connus sous ce nom à l'époque) étaient utilisés dans des casse-têtes mathématiques avant qu'ils soient connus des ingénieurs. L'ingénieur français Jean-Maurice-Émile Baudot (1845-1903) a utilisé les codes Gray pour la télégraphie en 1878. Il a reçu la légion d'honneur pour son travail. Il a donné son nom a une unité de mesure de la transmission télégraphique, appelé **baud**.

`chem.ch.huji.ac.il/~eugeniik/history/baudot.html`

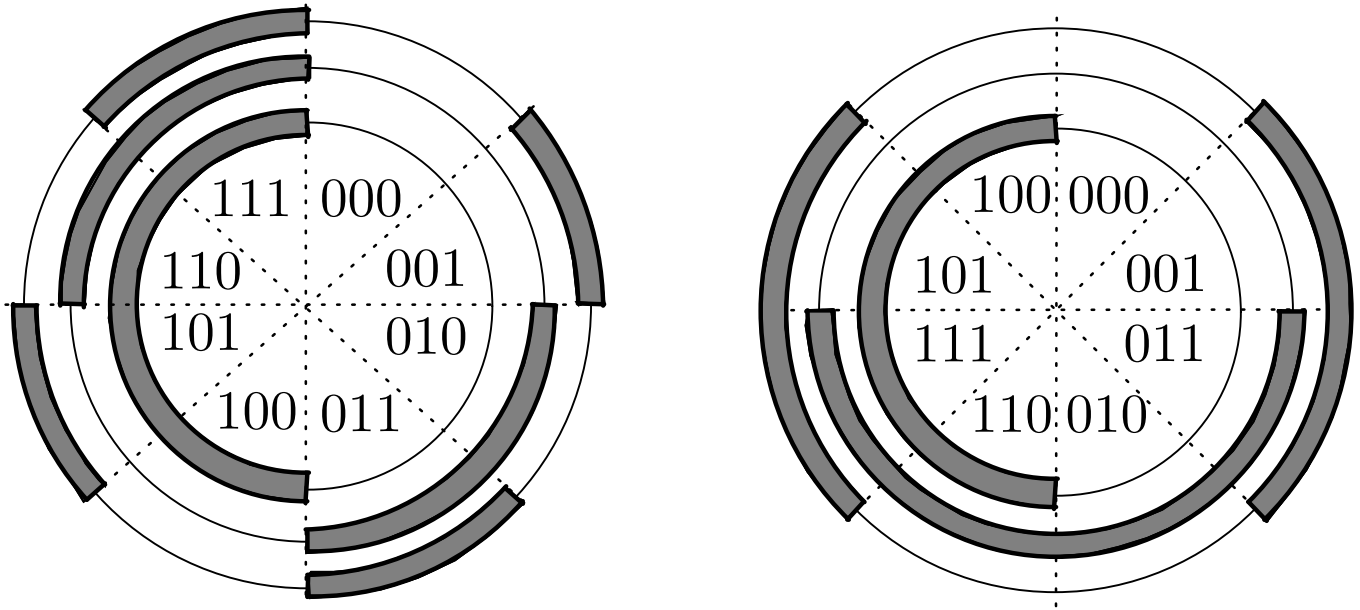
Note historique: Frank Gray

Un tube à vide utilisant le code Gray a été breveté par Frank Gray, un chercheur des laboratoires Bell, et a donné son nom au code.

F. Gray. "*Pulse code communication*", 17 mars 1953. U.S. patent no. 2,632,058.

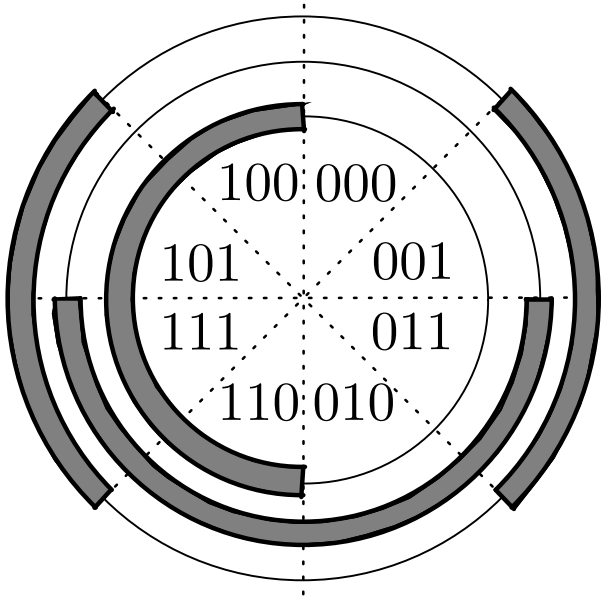
Un **code Gray** (ou code **binaire réfléchi**) est une manière d'étiqueter les secteurs d'un cercle de telle sorte que les secteurs adjacents soient étiquetés avec des chaînes binaires qui diffèrent d'exactement un bit.

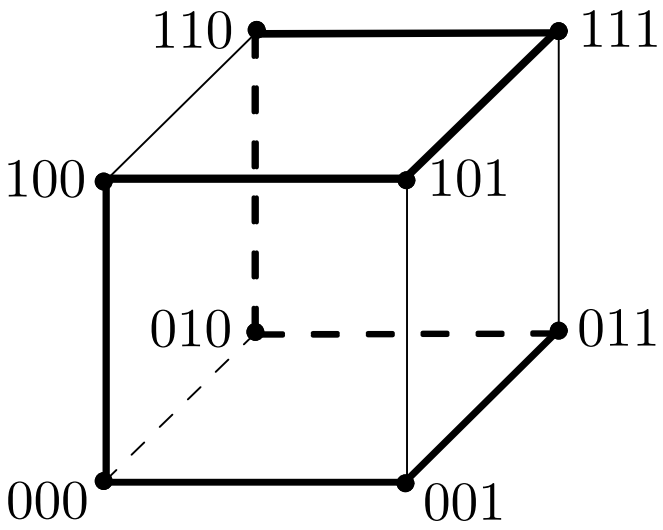
Code Gray



À gauche, une façon intuitive d'étiqueter huit secteurs d'un cercle, mais avec des secteurs adjacents qui diffèrent d'un, deux ou trois bits. À droite, le code Gray de telle sorte que les secteurs adjacents soient étiquetés avec des chaînes binaires qui diffèrent d'exactement un bit.

Code Gray





La recherche de chaînes binaires de longueur n, de telle façon que chacune diffère de la chaîne précédente exactement d'une position et que la dernière chaîne diffère de la première exactement d'une position également, est équivalent à chercher un cycle hamiltonien dans Q_n, le cube de dimension n.

Définition: graphe valué

Un **graphe valué** est un graphe dont les arcs sont affectés de valeurs.

Plus formellement, une graphe valué est un graphe $G = (V, E, w)$ constitué d'un ensemble V de sommets, d'un ensemble E d'arcs reliant ces sommets deux à deux et d'une fonction w de $V \times V$ dans $\mathbb{R}$. La fonction w associe à chaque pair de sommets un poids dans $\mathbb{R}$.

Note 1: Habituellement $w(u, v)$ est plus grand ou égal à zéro.
Note 2: On pose $w(u, v) = \infty$ si le couple de sommets (u, v) n'est pas un arc de E.

Définition: longueur du chemin

Soit le chemin c, de a à z, donné par la séquence d'arcs $e_1, e_2, ..., e_n$ du graphe, de telle sorte que $f(e_1) = (a, x_1)$, $f(e_2) = (x_1, x_2)$, ..., $f(e_n) = (x_{n-1}, z)$.

La **longueur** $L(c)$ **du chemin** c, dans un graphe valué, est la somme des valeurs attribuées aux arcs parcourus, c.-à-d.
$L(c) = w(a, x_1) + w(x_1, x_2) + \cdots + w(x_{n-1}, z)$.

Un problème courant consiste à trouver un chemin de longueur minimal entre deux sommets d'un graphe valué.

Note historique: Edsger Wybe Dijkstra

Rotterdam, 11 mai 1930 - 6 août 2002

Mathématicien et informaticien bien connu pour l'algorithme de chemin le plus court qui porte son nom. Il avait une aversion pour l'instruction `GOTO` en programmation et publia un article à ce sujet "*Go To Statement Considered Harmful*" en 1968. Il est aussi l'inventeur du langage `Algol`.

`www.cs.utexas.edu/users/EWD`

Algorithme du chemin minimal

Étant donnés une source a et une destination z, on va chercher un chemin μ de a à z minimisant $c(\mu)$.

Explication de la technique :
A chaque étape, on sélectionne un sommet x et on fixe la valeur du plus court chemin de a à x. On dispose de deux ensembles de sommets :
- *examinés* (noté S dans l'algorithme): C'est l'ensemble des sommets pour lesquels on connaît la valeur du plus court chemin issu de a.
- *à traiter* : C'est l'ensemble des sommets qui ont au moins un prédécesseur dans *examinés*.

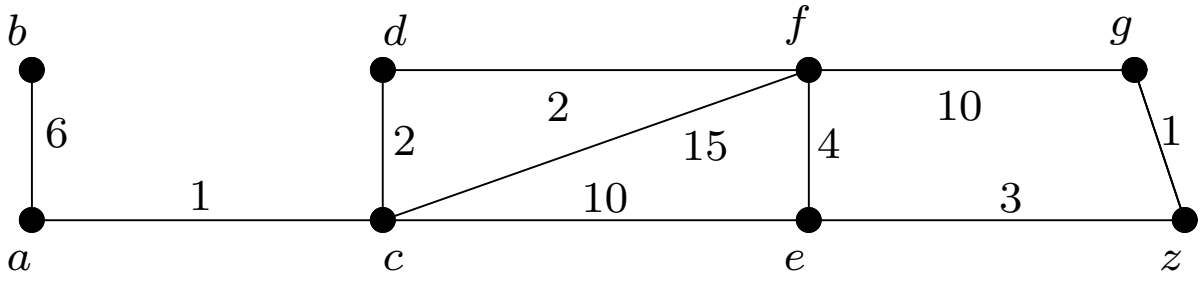
Pour les sommets x dans *à traiter*, on connaîtra la valeur d'un plus court chemin $\mu = (x_1, ..., x_k)$ avec $x_1 = s, x_k = x$ et $\{x_1, ..., x_k\} \in$ *examinés*. Le critère de sélection sera le suivant : on choisira dans *à traiter* le sommet x minimisant le coût de $\mu(a, x)$.

Algorithme du chemin minimal

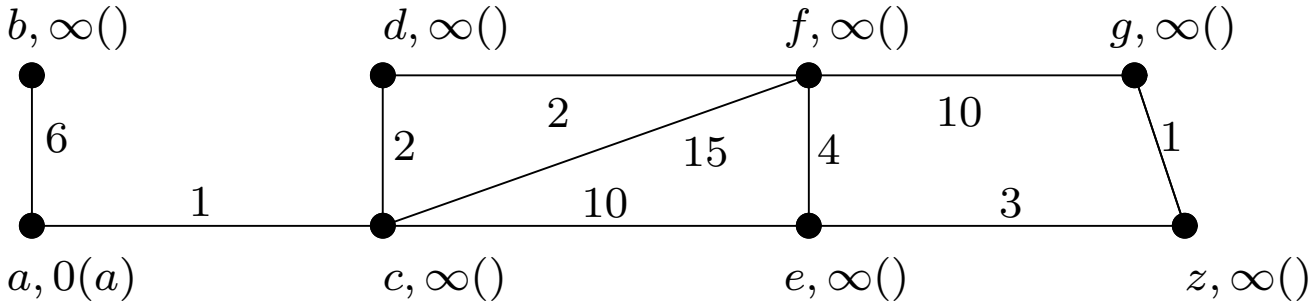
procédure *Dijkstra* (G: graphe simple connexe valué avec
toutes les valeurs positives.
a: sommet initial. z: sommet final)
$L(v) := \infty$ pour tout sommet v de G.
$L(a) := 0$
$S := \emptyset$
tant que $z \notin S$
début
$u :=$ un sommet non dans S avec $L(u)$ minimal.
$S := S \cup \{u\}$
pour tous les sommets v non dans S
si $L(u) + w(u, v) < L(v)$
alors $L(v) := L(u) + w(u, v)$
fin $\{L(z) =$ longueur du chemin minimal de a à $z\}$

Exemple de l'algorithme de Dijkstra, étape 1 de 8

Soit le graphe valué suivant. On cherche le chemin minimal entre a et z.

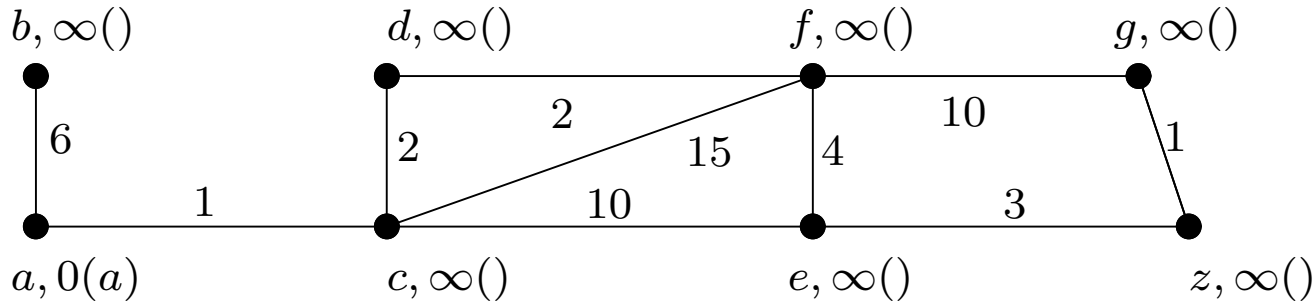


On initialise $L(v)$ à ∞ pour tout sommet v de G, $L(a)$ à 0 et S à $\emptyset$.

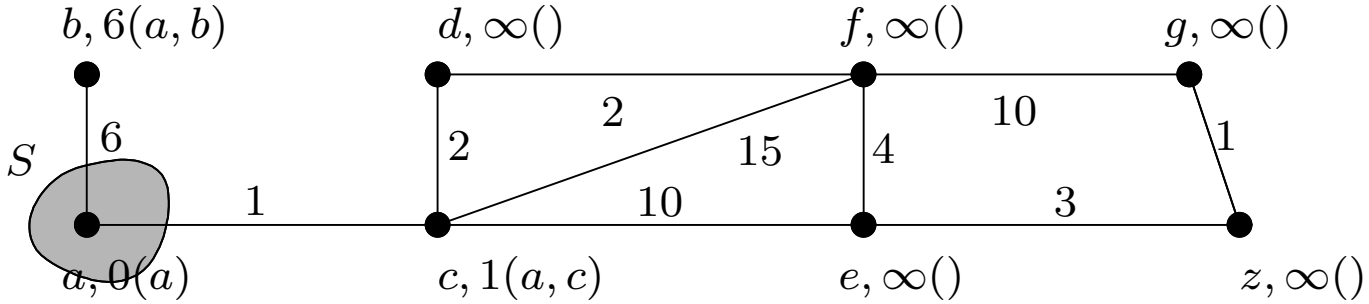


Exemple de l'algorithme de Dijkstra, étape 2 de 8

a est le sommet u non dans S tel que $L(u)$ est minimal.



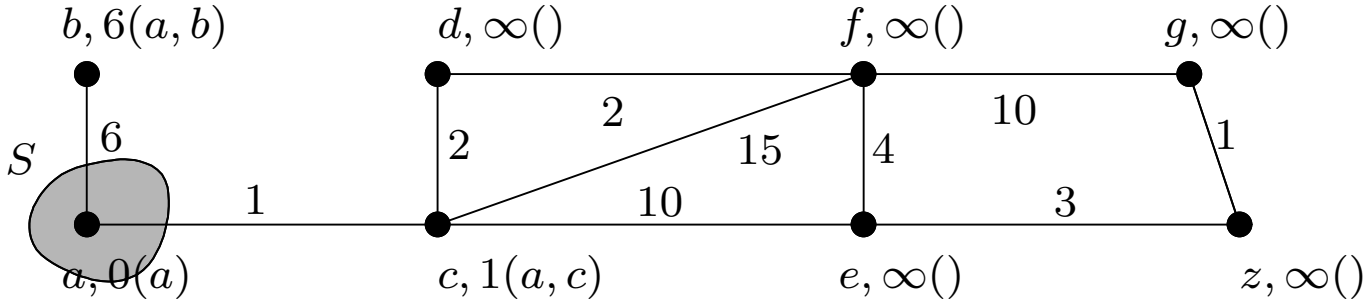
$S = S \cup \{a\}$ et $L(v)$ est mis à jour pour tout sommet v non dans S voisin de a.



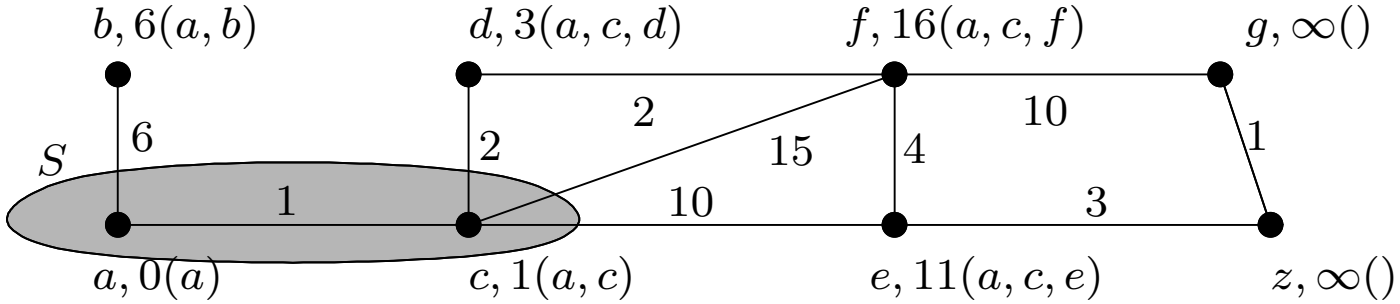
Note: Par manque d'espace, $L(a, c) = 1$, la longueur du chemin passant par les sommets a et c, est notée $1(a, c)$.

Exemple de l'algorithme de Dijkstra, étape 3 de 8

c est le sommet u non dans S tel que $L(u)$ est minimal.



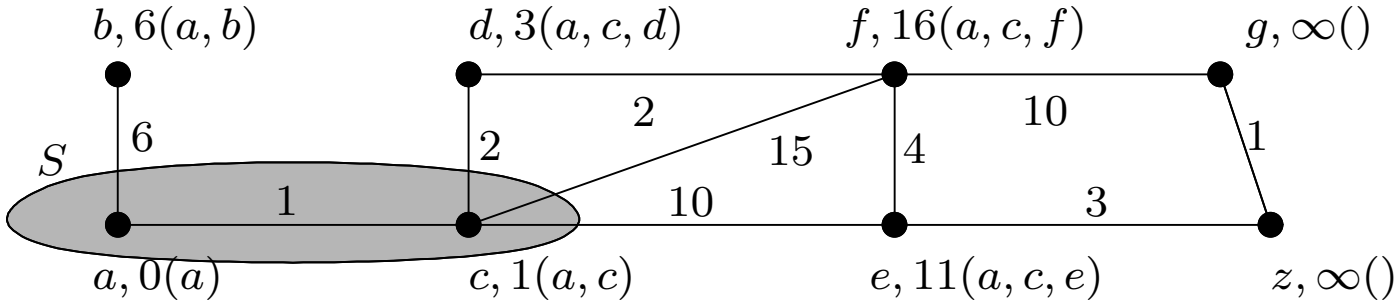
$S = S \cup \{c\}$ et $L(v)$ est mis à jour pour tout sommet v non dans S voisin de c.



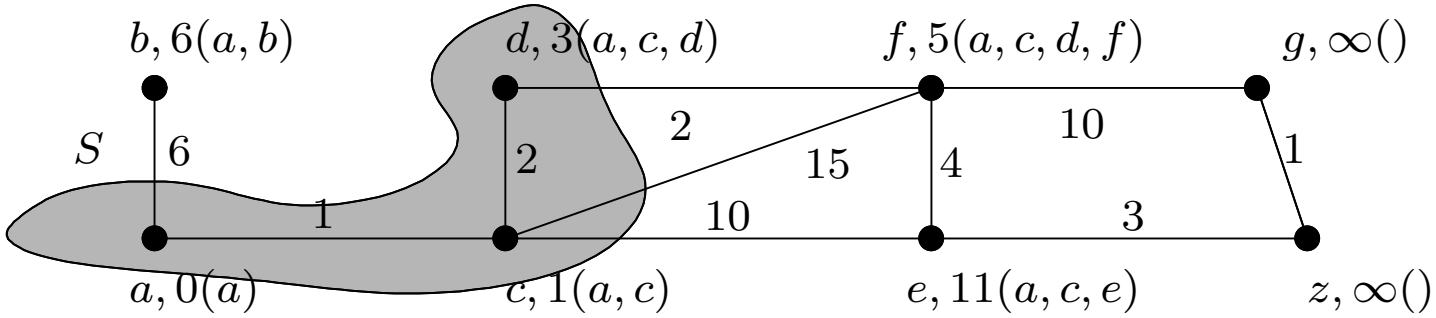
Note: Par manque d'espace, $L(a, c, d) = 3$, la longueur du chemin passant par a, c et d, est notée $3(a, c, d)$.

Exemple de l'algorithme de Dijkstra, étape 4 de 8

d est le sommet u non dans S tel que $L(u)$ est minimal.

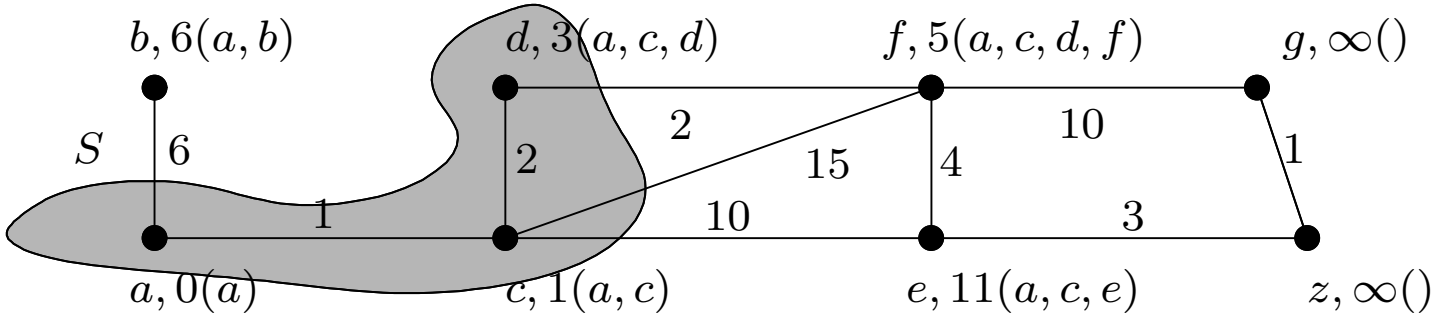


$S = S \cup \{d\}$ et $L(v)$ est mis à jour pour tout sommet v non dans S voisin de d.

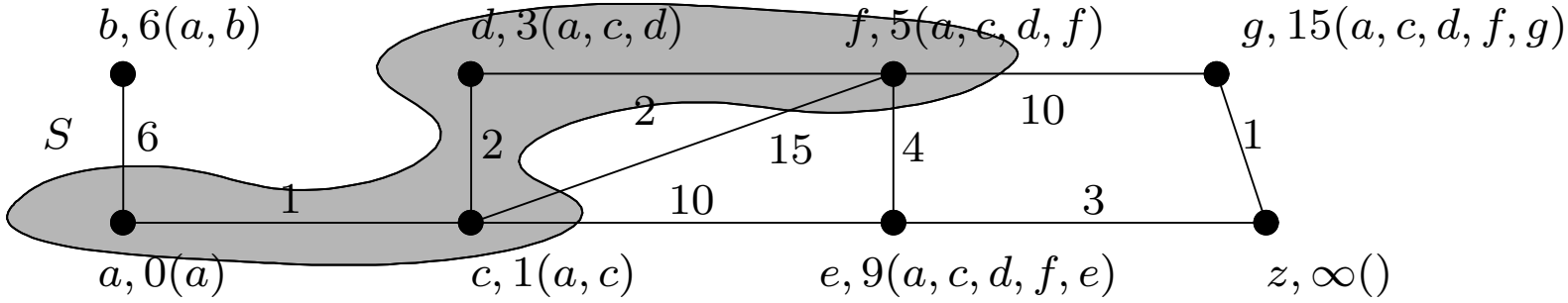


Exemple de l'algorithme de Dijkstra, étape 5 de 8

f est le sommet u non dans S tel que $L(u)$ est minimal.

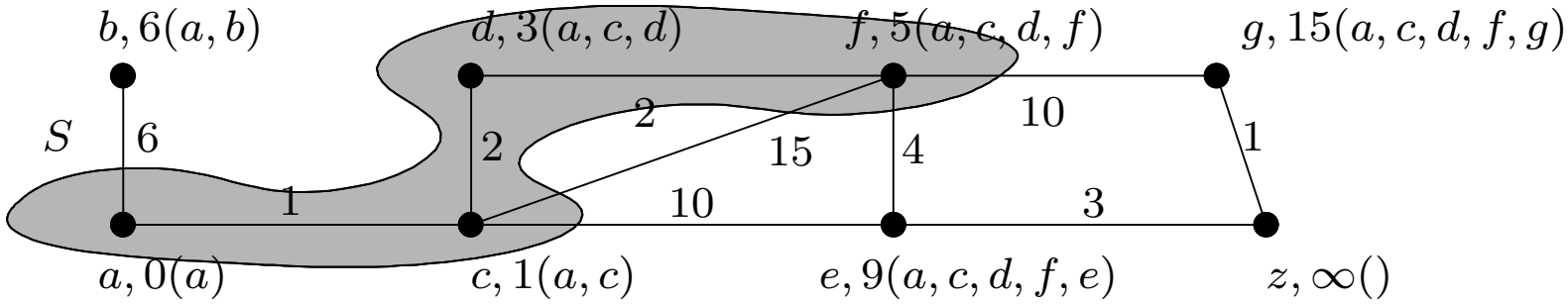


$S = S \cup \{f\}$ et $L(v)$ est mis à jour pour tout sommet v non dans S voisin de f.

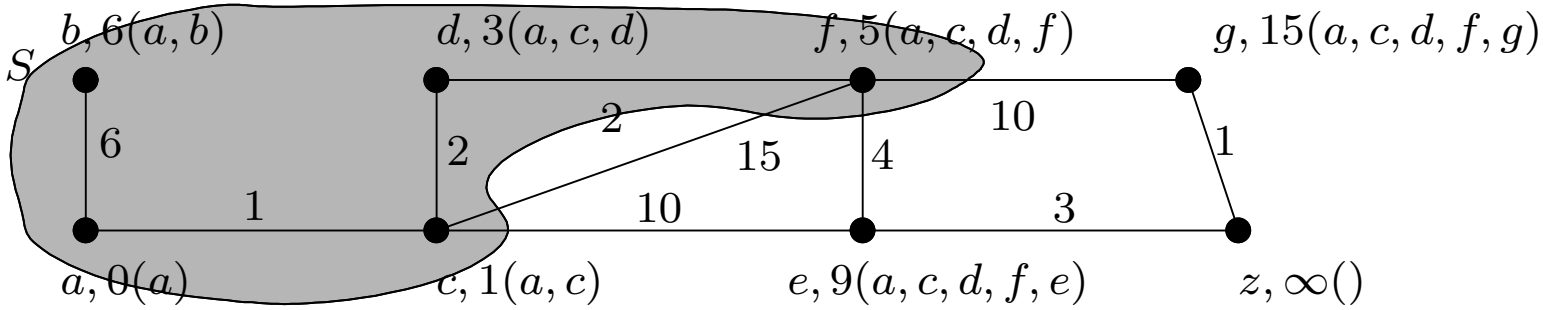


Exemple de l'algorithme de Dijkstra, étape 6 de 8

b est le sommet u non dans S tel que $L(u)$ est minimal.

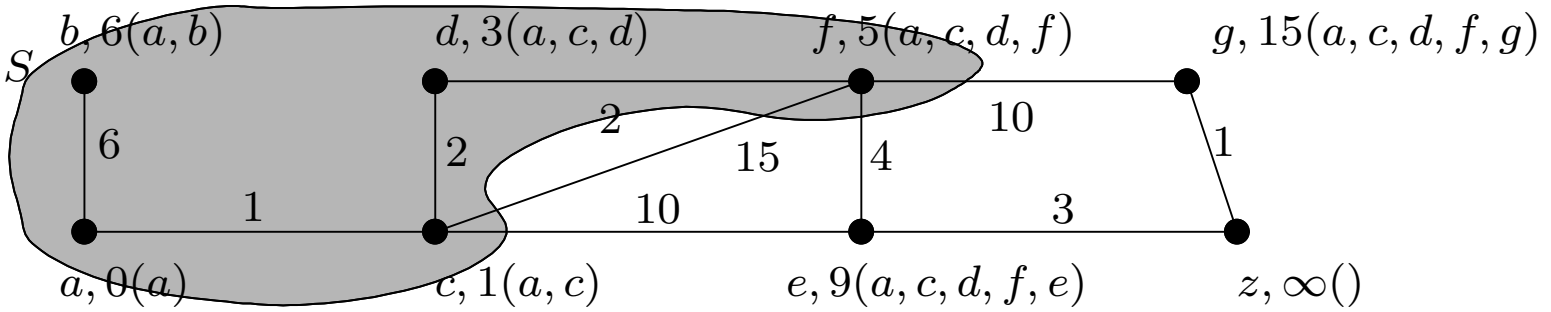


$S = S \cup \{b\}$ et $L(v)$ est mis à jour pour tout sommet v non dans S voisin de b (dans ce cas-ci, il n'y en a pas).

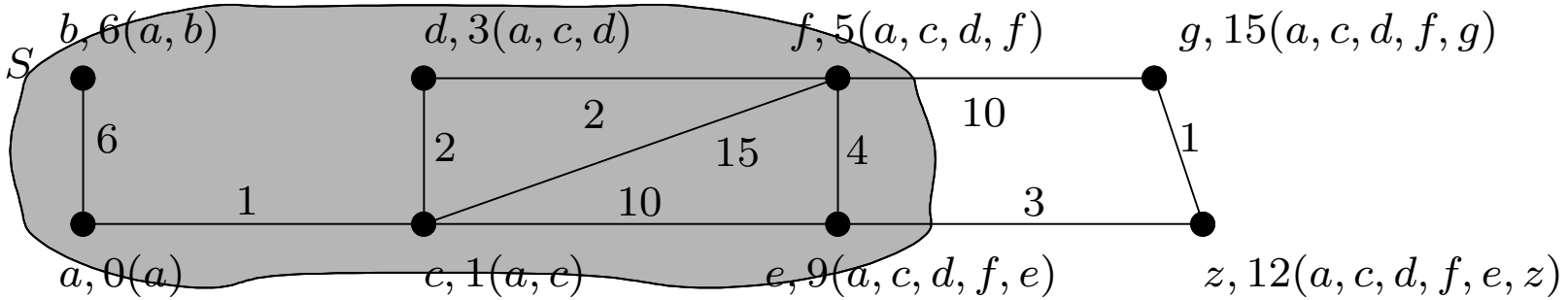


Exemple de l'algorithme de Dijkstra, étape 7 de 8

e est le sommet u non dans S tel que $L(u)$ est minimal.

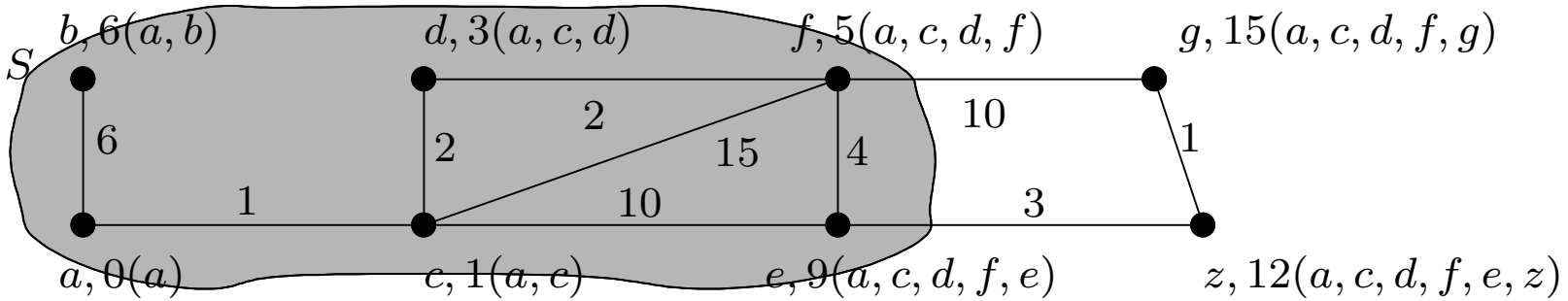


$S = S \cup \{e\}$ et $L(v)$ est mis à jour pour tout sommet v non dans S voisin de e.

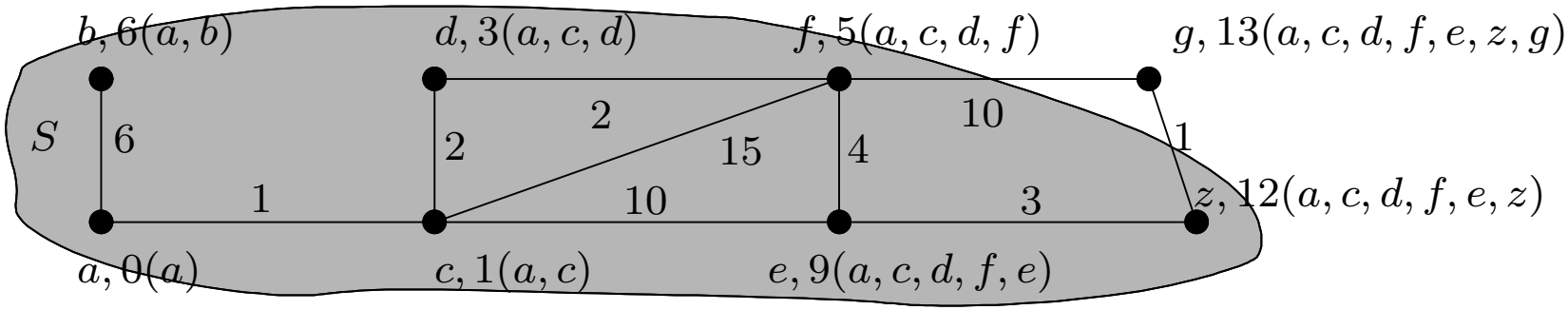


Exemple de l'algorithme de Dijkstra, étape 8 de 8

z est le sommet u non dans S tel que $L(u)$ est minimal.



$S = S \cup \{z\}$ et $L(v)$ est mis à jour pour tout sommet v non dans S voisin de z. L'algorithme s'arrête parce que $z \in S$.



Étude de l'algorithme de Dijkstra

Terminaison

THÉORÈME: L'algorithme se termine toujours en n étapes ou moins pour un graphe ayant n sommets.

Exactitude

THÉORÈME: L'algorithme de Dijkstra permet de trouver la longueur du chemin minimal entre deux sommets dans un graphe valué non orienté simple et connexe.

Preuve par récurrence sur le nombre de passages dans la boucle **while**.

Complexité de l'algorithme de Dijkstra

Ordre de complexité

THÉORÈME: L'algorithme de Dijkstra utilise $O(n^2)$ opérations (additions et comparaisons) pour trouver la longueur du chemin minimal entre deux sommets dans un graphe valué non orienté simple et connexe. ($O(n)$ si on utilise une liste d'adjacence).

PREUVE : L'initialisation est en O(n).

Pour trouver x dans le 'tant que' tel que $c(x)$ minimum, on parcourt un tableau jusqu'à trouver la plus petite valeur. C'est en $O(n)$.

Le traitement d'un sommet x impose de regarder tous ses voisins, ce qui se fait en $O(d^+(x))$ où d^+ est le nombre de degrés sortants de x.

Le coût total est donc $\sum_{x \in X} O(n + d^+(x)) = O(n^2 + m) = O(n^2)$

Algorithme de Floyd

L'algorithme de Floyd trouve la longueur du chemin minimal entre toutes les paires de sommets dans un graphe simple connexe valué. Cependant, il ne permet pas de construire les chemin minimaux eux-mêmes.

Robert W. Floyd. "*Algorithm 97: Shortest path*", Comm ACM, vol 5 (juin 1962), p. 345 et suivantes.

Algorithme de Floyd

procédure *Floyd* (G: graphe simple valué
de sommets $v_1, v_2, ..., v_n$)
pour $i := 1$ **à** n
pour $j := 1$ **à** n
$d(v_i, v_j) = w(v_i, v_j)$
$d(v_i, v_j) = \infty$ si $\{v_i, v_j\}$ n'est pas un arc
pour $i := 1$ **à** n
pour $j := 1$ **à** n
pour $k := 1$ **à** n
si $d(v_j, v_i) + d(v_i, v_k) < d(v_j, v_k)$
alors $d(v_j, v_k) := d(v_j, v_i) + d(v_i, v_k)$
$\{d(v_i, v_j)$ est la longueur du chemin minimal entre v_i et v_j

Algorithme de Floyd - Principe

Description de l'algorithme : L'algorithme repose sur la remarque suivante : si $(a_0, ..., a_i, ..., a_p)$ est un plus court chemin de a_0 à a_p , alors $(a_0, ..., a_i)$ est un plus court chemin de a_0 à a_i , et $(a_i, ..., a_p)$ un plus court chemin de a_i à a_p . De plus, comme les arêtes sont valuées positivement, tout chemin contenant un cycle est nécessairement plus long que le même chemin sans le cycle, si bien qu'on peut se limiter à la recherche de plus courts chemins passant par des sommets deux à deux distincts.

Floyd montre donc qu'il suffit de calculer la suite de matrices définies par :

$$M_{i,j}^{(k)} = min(M_{i,j}^{(k-1)}, M_{i,k}^{(k-1)} + M_{k,j}^{(k-1)})$$

Chaque matrice se calculant en $O(n^2)$, l'algorithme final est en $O(n^3)$.

Module Arbres

Introduction aux arbres

Chapitre .1 Terminologie (128)

- Arbre (128) • Forêt (129)
- Racine et arborescence (131)
- Père, fils, frère, feuille, sommet interne (133)
- Sous-arbre (135)
- Arbre m-aire (136)
- Arbre quaternaire et octaire (137)
- Arborescence ordonnée (138)
- Propriété des arbres (140)
- Niveau et hauteur (143)

Suite page suivante...

Module Arbres

...suite de la page précédente

Chapitre .2 Parcours d'arbres (148)

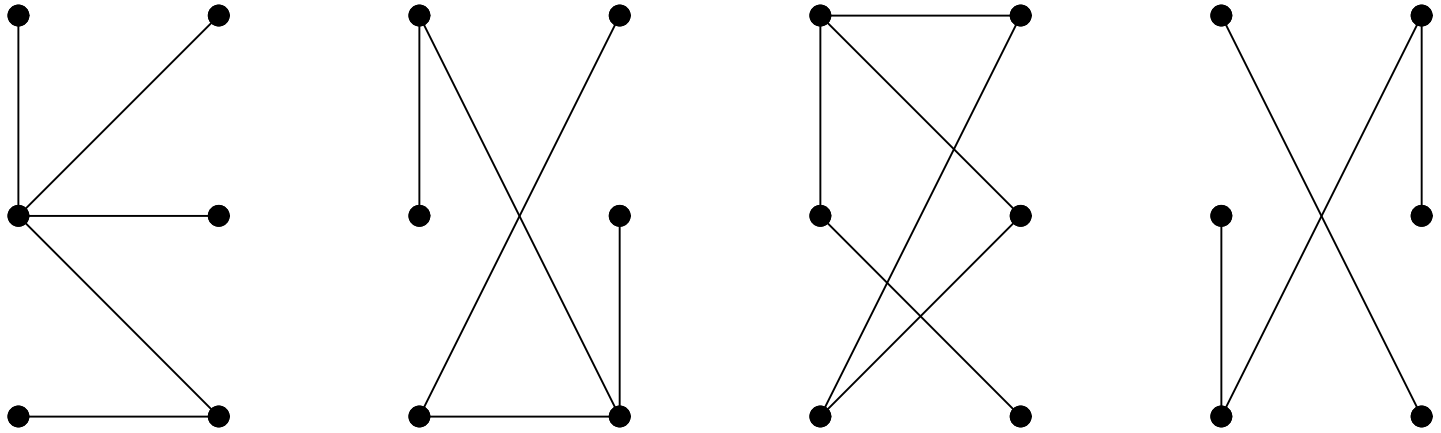
- Système d'adressage universel (149)
- Parcours préfixe (153)
- Parcours infixe (156)
- Parcours postfixe (159)
- Parcours d'expressions arithmétiques et arborescence (166)
- Forme infixe d'une expression (169)
- Forme préfixe d'une expression (171)
- Forme postfixe d'une expression (176)

Définition: arbre

Un **arbre** est un graphe connexe non orienté sans cycle.

Comme un arbre ne peut pas contenir de cycle, alors il ne contient pas d'arcs multiples ni de boucles. Donc tout arbre est un graphe simple.

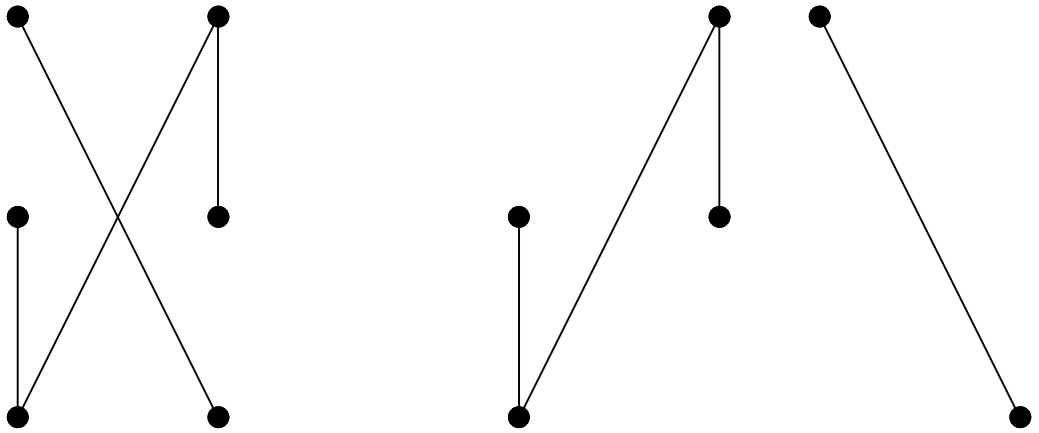
Le premier et le deuxième sont des arbres, le troisième et el quatrième n'en sont pas.



Définition: forêt

Une **forêt** est un graphe non orienté sans cycle.

La différence entre un arbre et une forêt est que dans une forêt, la connexité n'est pas imposée. Si chacune des composantes connexes d'un graphe non connexe est un arbre, alors le graphe est une forêt.



Définition équivalente: arbre

THÉORÈME: Un graphe non orienté est un arbre si et seulement si chaque paire de sommets est reliée par une chaîne simple et unique.

PREUVE : $\Rightarrow$) Soit T est un arbre, c-à-d un graphe simple, connexe sans cycles. Soient x et y 2 sommets de T. Puisque T est connexe, il existe un chemin simple entre x et y (Th. du transparent 72). De plus ce chemin est unique car s'il y en avait 2 ils pourraient constituer un cycle.
$\Leftarrow$) Si chaque paire de sommets est reliée par un chemin simple et unique, alors T connexe. Si T contenait un cycle simple, il y aurait 2 chemins possibles entre 2 sommets appartenant à ce cycle. T est donc connexe sans cycle, c'est un arbre.)

Définition: racine et arbores-cence

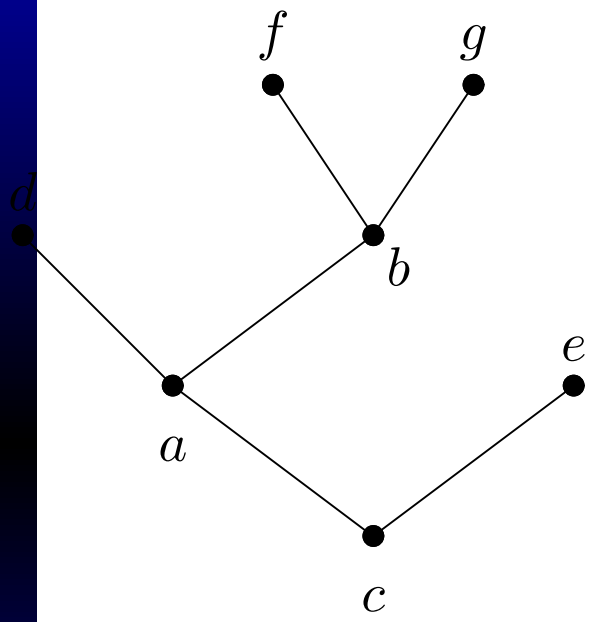
Dans un arbre, la **racine** est un sommet particulier.

Le choix du sommet comme racine est arbitraire.

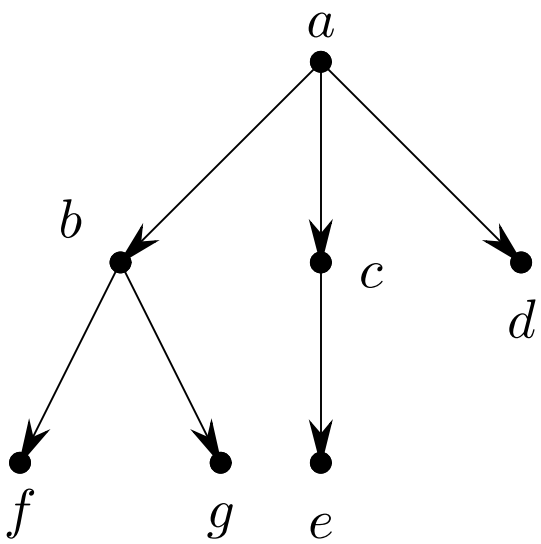
L'arbre combiné avec sa racine produit un graphe orienté qu'on appelle une **arborescence**.

On dessine habituellement une arborescence avec la racine en haut du graphe.

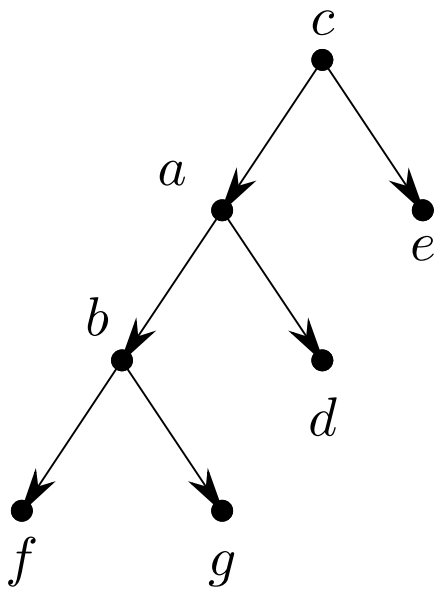
Définition: racine et arbores-cence



Arbre



Arborescence
avec a comme racine

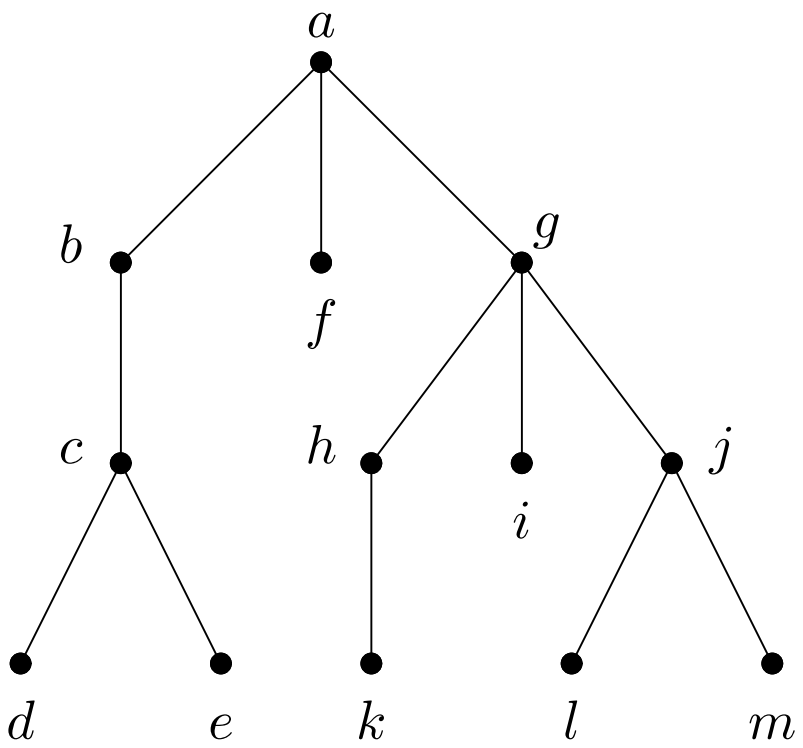


Arborescence
avec c comme racine

Définitions dans une arborescence

- Si v est un sommet d'une arborescence T différent de la racine, le **père** de v est le sommet unique u tel qu'il y a un arc orienté unique de u à v.
- Lorsque u est le père de v, alors v s'appelle le **fils** de u.
- Les sommets ayant le même père s'appellent les **frères**.
- Les **ancêtres** d'un sommet différent de la racine sont les sommets du chemin partant de la racine à ce sommet, excluant ce sommet et incluant la racine.
- Les **descendants** d'un sommet v sont les sommets qui ont v comme ancêtre.
- Un sommet d'un arbre est une **feuille** s'il n'a pas de fils.
- Les sommets qui ont des fils sont des **sommets internes**.

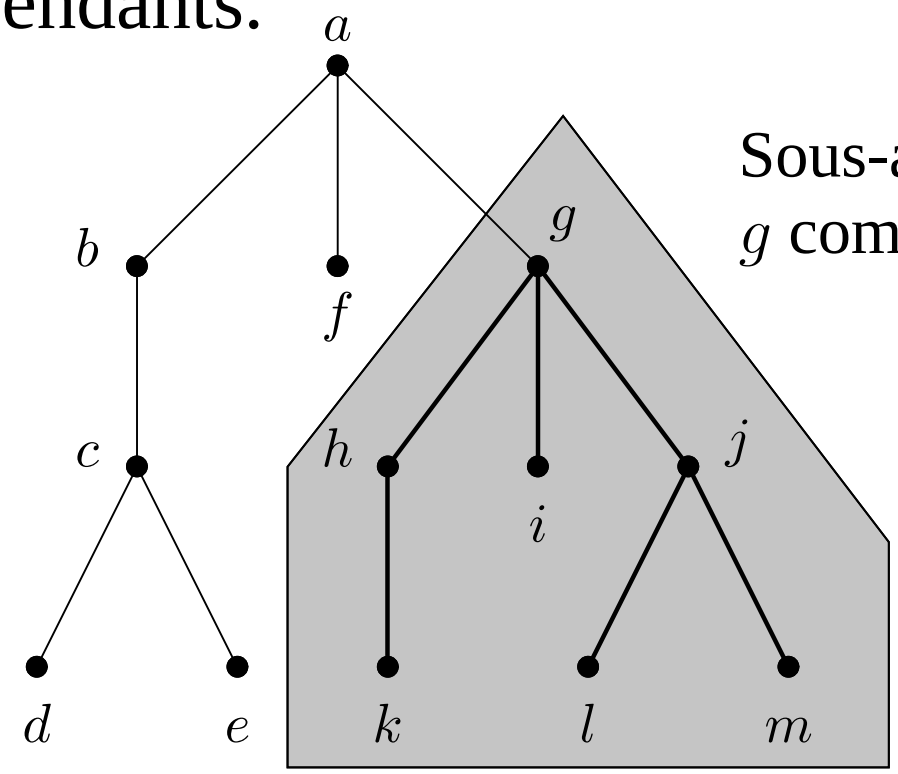
Exemple d'une arborescence



- Le père de c est b.
- Les fils de g sont h, i et j.
- Les frères de h sont i et j.
- Les ancêtres de e sont c, b et a.
- Les descendants de b sont c, d et e.
- Les sommets internes sont a, b, c, g, h et j.
- Les feuilles sont d, e, f, i, k, l et m.

Définition: sous-arbre

Si a est un sommet d'un arbre, un **sous-arbre** ayant a comme racine est le sous-graphe de l'arbre constitué de a, de ses descendants et de tous les arcs incidents à ces descendants.



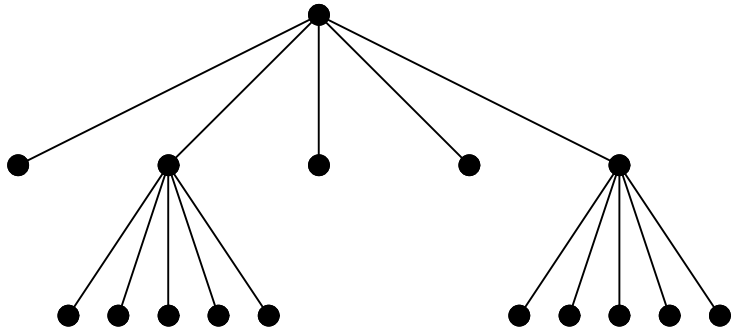
Sous-arbre ayant g comme racine

Définition: arbre m-aire

Un **arbre m-aire** est une arborescence où chaque sommet interne n'a pas plus de m fils.

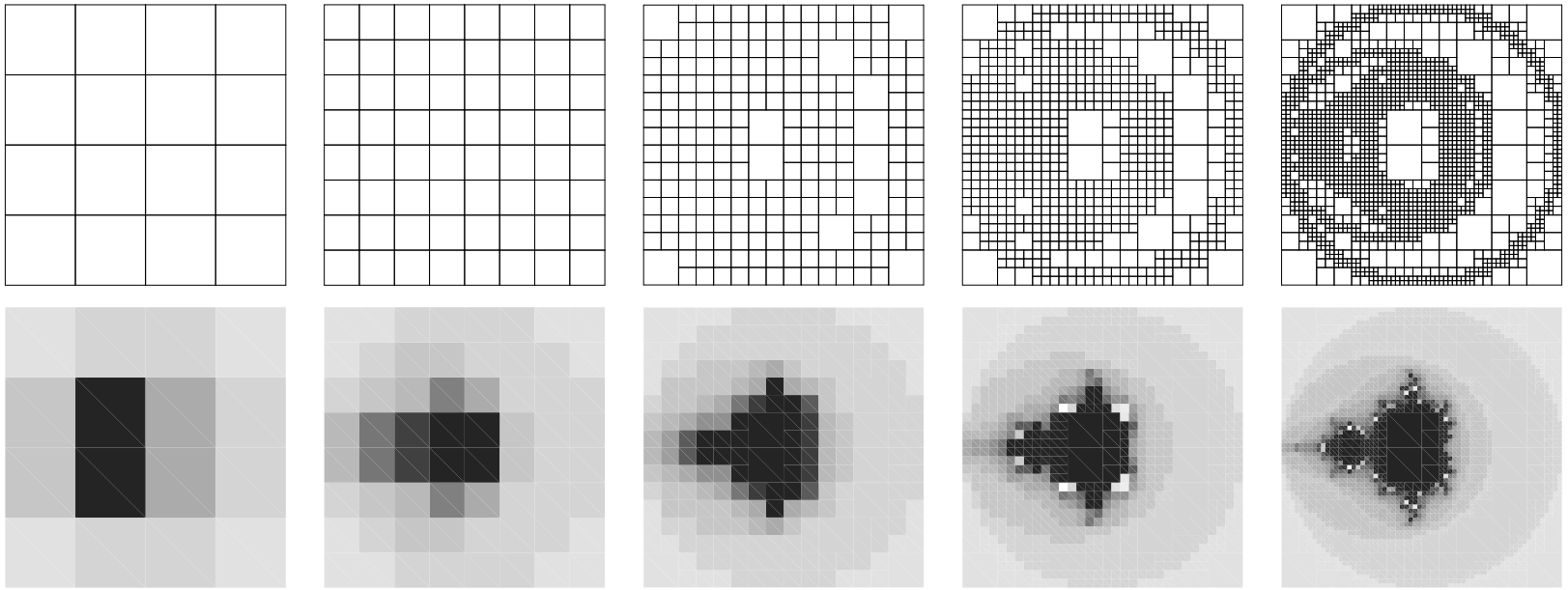
Un **arbre binaire** est un arbre m-aire avec $m = 2$.

L'arbre s'appelle un **arbre m-aire complet** si chaque sommet interne a exactement m fils.



Définition: arbre quaternaire et octaire

L'**arbre quaternaire** (**quadtree**) est un arbre 4-aire com-
plet. L'**arbre octaire** (**octree**) est un arbre 8-aire complet. Les arbres quaternaires et octaires sont utilisés, en 2D et 3D respectivement, pour localiser un point dans l'espace ou pour représenter un espace de façon plus efficace.



Fractale de Mandelbrot sur un quadtree de niveau 2, 3, 4, 5 et 6.

Définition: arborescence ordonnée

Une **arborescence ordonnée** est une arborescence dans laquelle les fils de chaque sommet interne sont ordonnés. Par convention, l'ordre va de gauche à droite.

Dans un arbre binaire ordonné, si un sommet interne a deux fils, le premier fils s'appelle le **fils de gauche**, et le deuxième, le **fils de droite**.

L'arbre dont la racine est le fils gauche d'un sommet s'appelle le **sous-arbre gauche** de ce sommet, et l'arbre dont la racine est le fils droit d'un sommet s'appelle le **sous-arbre droit** de ce sommet.

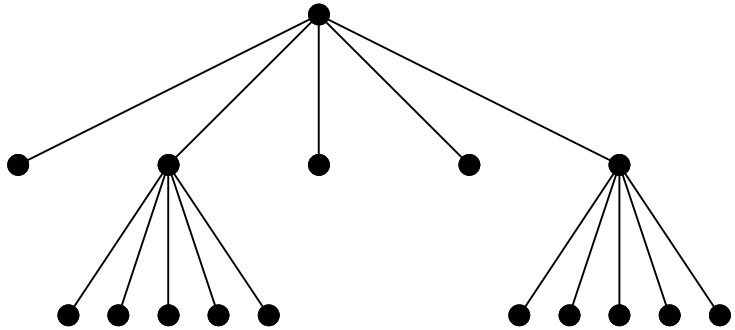
Modèles utilisant des arbres

- Arbre généalogique
- Représentation des molécules
- Arborescence de la direction d'une compagnie
- Système de fichiers
- Réseau d'ordinateurs
- Arbre de décision
- Arbre de tri
- Arbre de localisation
- Etc...

Propriété des arbres

THÉORÈME: Un arbre à n sommets comporte $n - 1$ arcs.

PREUVE On choisit le sommet r comme racine de l'arbre. On établit une injection entre les arcs et les sommets autres que r, en associant le sommet final d'un arc à cet arc. Comme il y a $n - 1$ sommets autres que r, il y a $n - 1$ arcs.

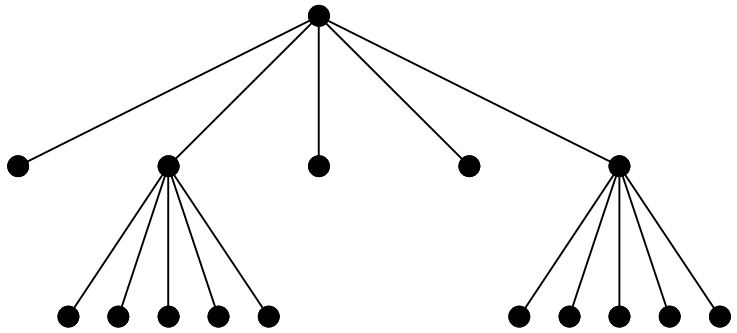


Propriété des arbres

THÉORÈME: Un arbre m-aire complet ayant i sommets internes contient $n = m\,i + 1$ sommets.

PREUVE

Chaque sommet sauf la racine est fils d'un sommet interne. Comme chacun des i sommets a m fils, il y a $mi + 1$ sommets dans l'arbre (on rajoute la racine).



Propriété des arbres

Soient un arbre m-aire complet, n le nombre de sommets, i le nombre de sommets internes et l le nombre de feuilles. Si on connaît une seule des trois inconnues n, i ou l, on peut déduire les deux autres.

THÉORÈME: Un arbre m-aire complet ayant

i) n sommets comporte $i = (n-1)/m$ sommets internes et $l = (n(m-1)+1)/m$ feuilles,

ii) i sommets internes comporte $n = mi + 1$ sommets et $l = i(m-1)+1$ feuilles,

iii) l feuilles comporte $n = (ml-1)/(m-1)$ sommets et $i = (l-1)/(m-1)$ sommets internes.

PREUVE en exercice

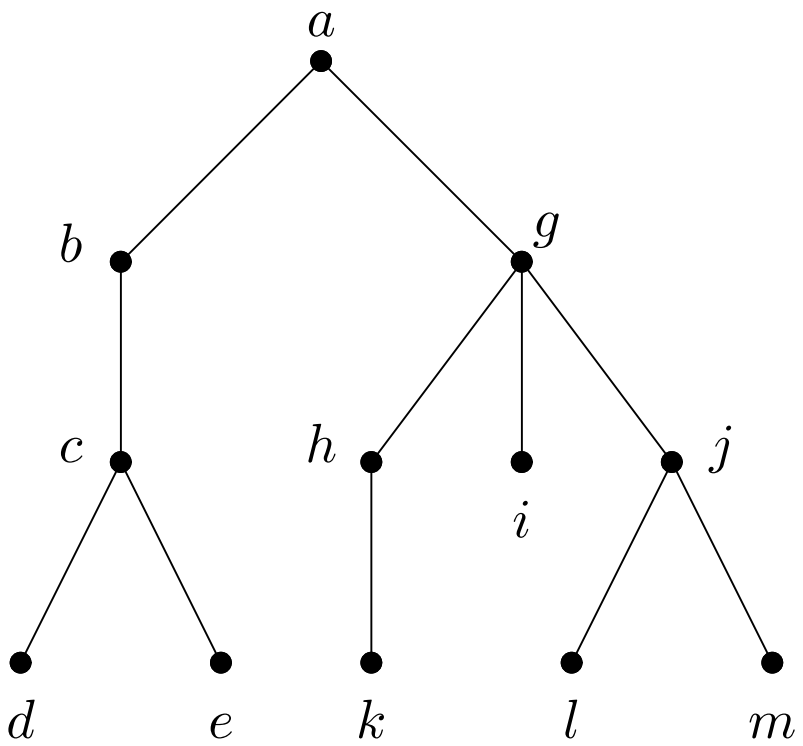
Définition: niveau et hauteur

Le **niveau** d'un sommet v dans une arborescence est égal à la longueur du chemin unique entre la racine et ce sommet.

Le niveau de la racine est par définition égal à zéro.

La **hauteur** ou **profondeur** d'une arborescence est le maximum des niveaux des sommets.

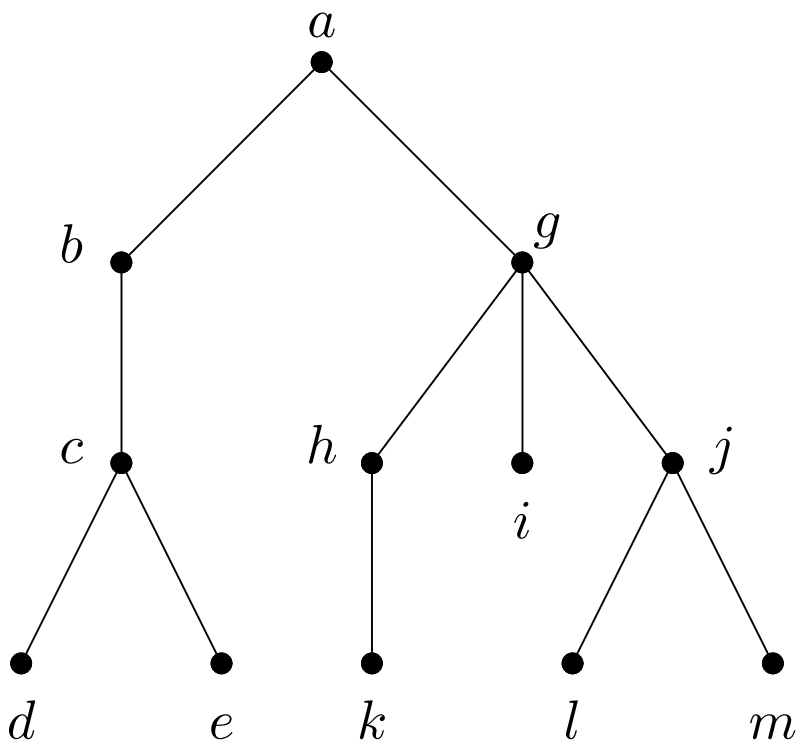
Exemple: niveau et hauteur



- La racine a est au niveau 0.
- Les sommets b et g sont au niveau 1.
- Les sommets c, h, i et j sont au niveau 2.
- Les sommets d, e, k, l et m sont au niveau 3.
- La hauteur de cet arborescence est 3.

Définition: arbre équilibré

Une arborescence m-aire de hauteur h est **équilibrée** si toutes les feuilles sont aux niveau h ou $h-1$.



Propriété des arbres (1)

THÉORÈME: Il y a au plus m^h feuilles dans un arbre m-aire de hauteur h.

PREUVE

Par induction sur la hauteur.

- Un arbre m-aire de hauteur 1 contient au plus m feuilles.

- Supposons le résultat vrai pour les arbres m-aires de hauteur inférieure à h.

Soit une arbre T de hauteur h. Les feuilles de T sont les feuilles des sous-arbres obtenus en enlevant la racine de T. Il y en a au plus m et chacun est de hauteur inférieure ou égale à $h-1$, donc chacun contient au plus m^{h-1} feuilles. Donc T contient au plus $m.m^{h-1} = m^h$ feuilles.

Propriété des arbres (2)

COROLLAIRE: Si un arbre m-aire de hauteur h comporte l feuilles, alors $h \geq \lceil \log_m l \rceil$. Si l'arbre m-aire est complet et équilibré, alors $h = \lceil \log_m l \rceil$.

PREUVE

On a $l \leq m^h$. Par le logarithme en base m on obtient $\log_m l \leq h$ et donc $h \geq \lceil \log_m l \rceil$.

Puisque l'arbre est équilibré, chaque feuille est au niveau h ou $h-1$ et il y a au moins une feuille au niveau h. Il s'ensuit qu'il doit y avoir plus de m^{h-1} feuilles (à prouver). Puisque $l \leq m^h$, on a $m^{h-1} < l \leq m^h$. Par le logarithme en base m on obtient $h - 1 < \log_m l \leq h$ et donc $h = \lceil \log_m l \rceil$.

Motivation du problème du parcours d'arbres

Les arborescences ordonnées sont utilisées pour stocker des données. On veut pouvoir parcourir chaque sommet d'une arborescence ordonnée pour accéder aux données.

Dans une arborescence ordonnée, les fils d'un sommet interne apparaissent de gauche à droite représentant l'ordre partiel entre ces fils.

Trouver une manière de parcourir tous les sommets d'une arborescence ordonnée consiste à construire un ordre total compatible des sommets.

Système d'adressage universel

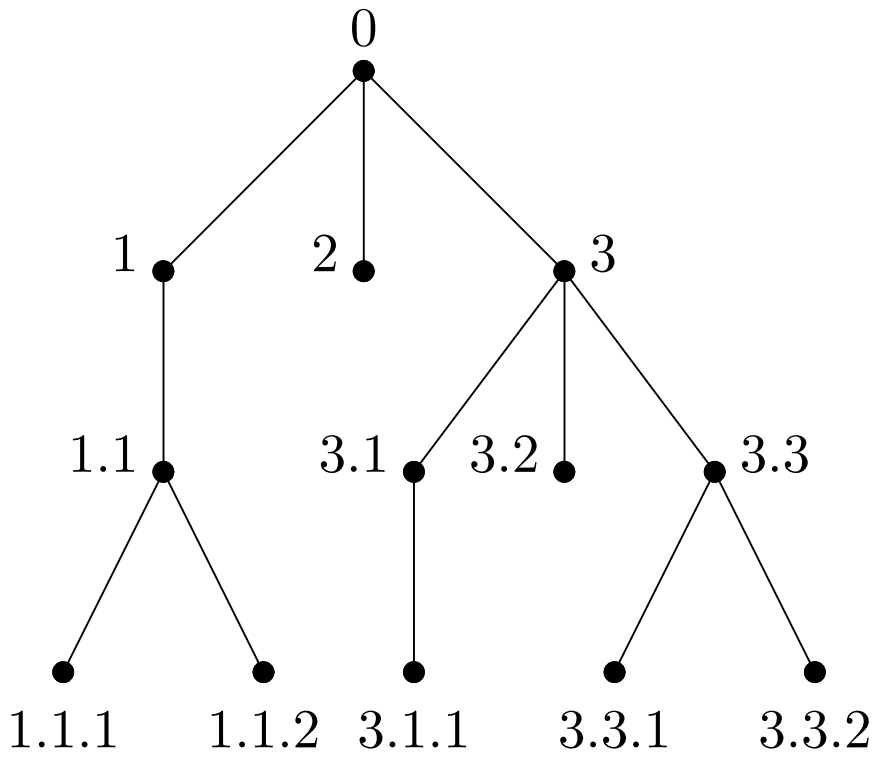
On étiquette récursivement les sommets d'une arborescence comme suit:

1. On étiquette la racine avec l'entier 0. Ensuite, on étiquette ses k fils (au niveau 1) de gauche à droite avec $1, 2, 3, ..., k$.

2. Pour chaque sommet v au niveau n avec l'étiquette A, on étiquette les k_v fils, de gauche à droite, avec $A.1, A.2, ..., A.k_v$.

En suivant cette procédure, un sommet v au niveau n, pour $n \geq 1$, est étiqueté $x_1.x_2.\cdots .x_n$, où le chemin unique de la racine à v passe par le x_1-ième sommet au niveau 1, le x_2-ième sommet au niveau 2, etc.

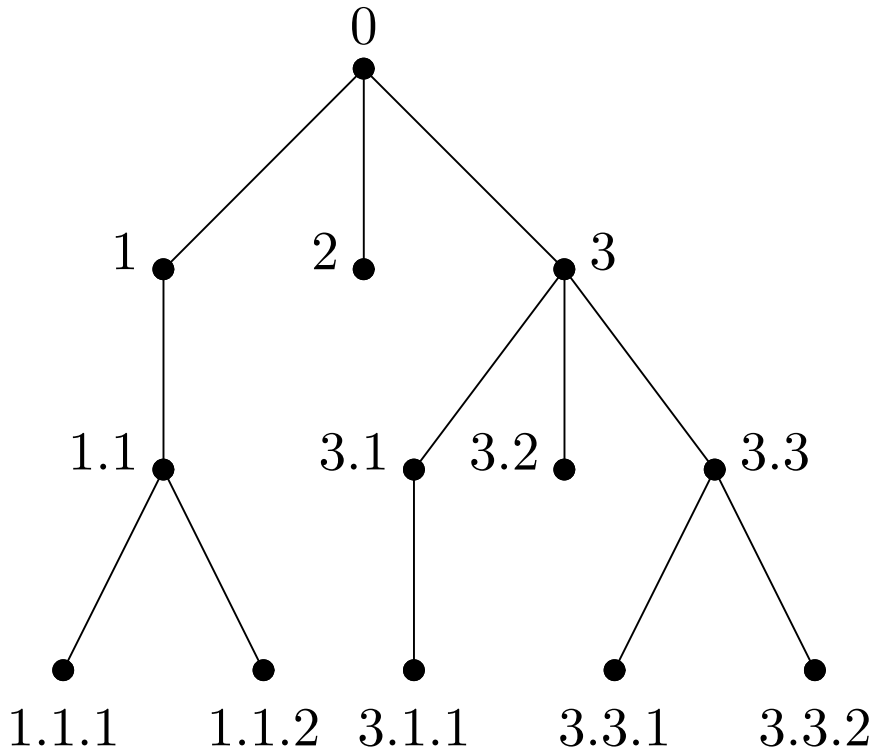
Exemple du système d'adressage universel



Ordre lexicographique

On peut ordonner totalement les sommets en utilisant l'ordre lexicographique de leurs étiquettes dans le système d'adressage universel. Le sommet étiqueté $x_1.x_2.\cdots.x_n$ est plus petit que le sommet étiqueté $y_1.y_2.\cdots.y_m$ s'il existe un i, $0 \leq i \leq n$ tel que $x_1 = y_1$, $x_2 = y_2$, ..., $x_{i-1} = y_{i-1}$ et $x_i < y_i$; ou si $n < m$ et $x_i = y_i$ pour $i = 1, 2, ..., n$.

Exemple du système d'adressage universel



Ordre lexicographique des sommets de l'arborescence:

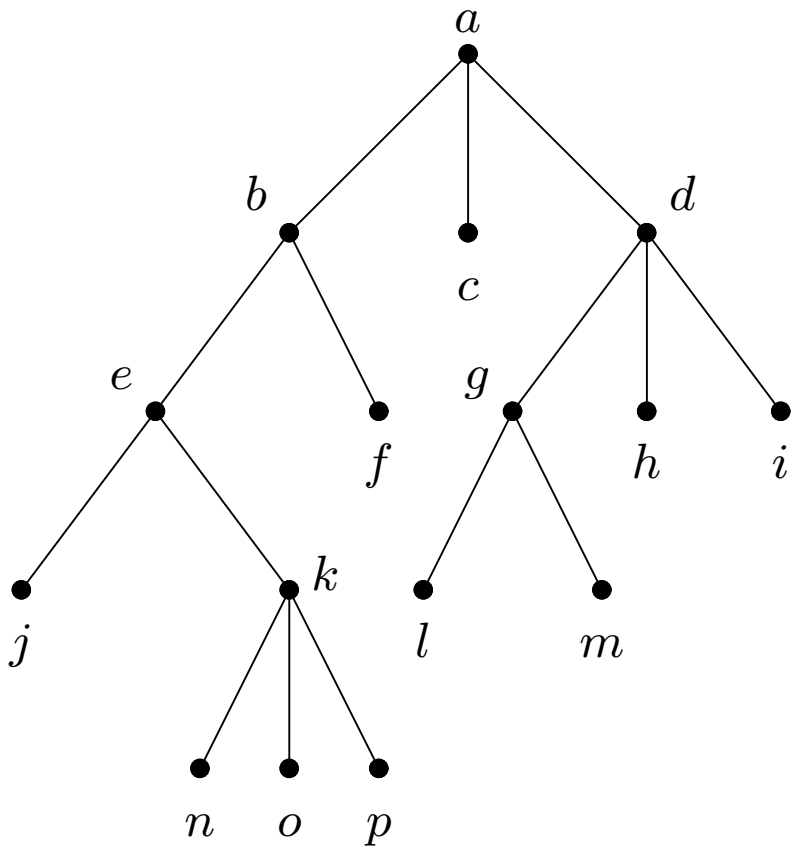
$0 < 1 < 1.1 < 1.1.1 < 1.1.2 < 2 < 3 < 3.1 < 3.1.1 < 3.2 < 3.3 < 3.3.1 < 3.3.2$

Définition: parcours préfixe

Soit T une arborescence ordonnée avec la racine r. Si T est constituée uniquement de r, alors r est le **parcours préfixe** de T. Sinon, on suppose que T_1, T_2, ..., T_n sont les sous-arbres de r de gauche à droite dans T. Le **parcours préfixe** débute en parcourant r. Il se poursuit avec le parcours préfixe de T_1, puis le parcours préfixe de T_2 et ainsi de suite jusqu'à ce que le parcours préfixe de T_n soit effectué.

Le parcours préfixe d'une arborescence ordonnée produit le même ordre que les sommets de l'ordre obtenu en utilisant le système d'adressage universel.

Exemple de parcours préfixe



Parcours préfixe: $a\,b\,e\,j\,k\,n\,o\,p\,f\,c\,d\,g\,l\,m\,h\,i$

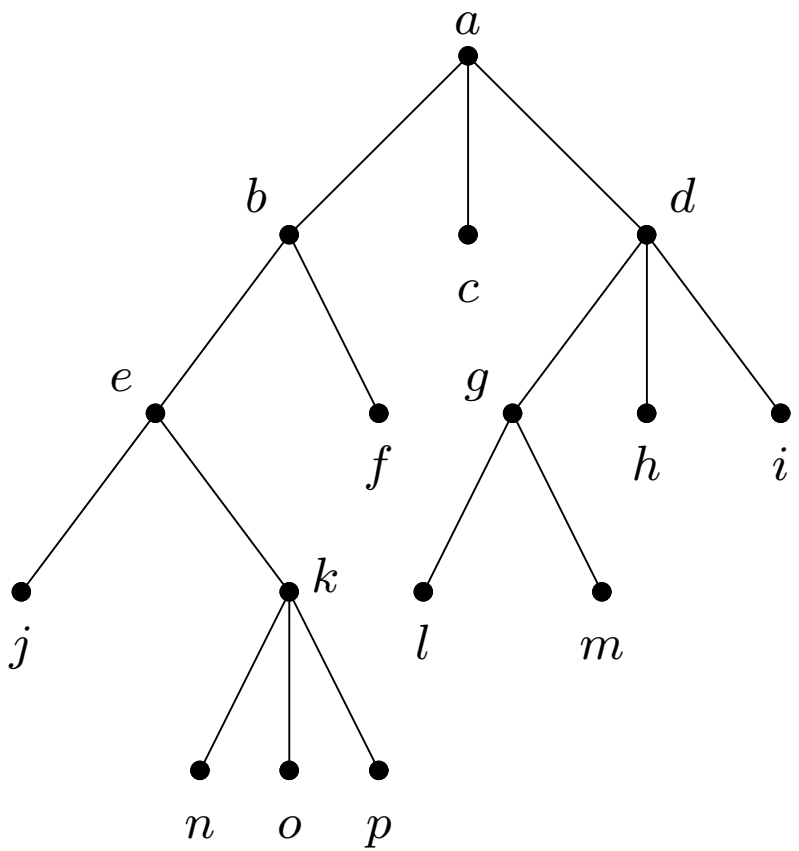
Algorithme: Parcours préfixe

procédure *préfixe* (T: arborescence ordonnée)
$r :=$ racine de T
ranger r dans la liste
pour chaque fils c de r de gauche à droite
début
    $T(c) :=$ sous-arbre avec c comme racine
    préfixe$(T(c))$
fin {liste contient le parcours préfixe de T}

Définition: parcours infixe ou symétrique

Soit T une arborescence ordonnée avec la racine r. Si T est constituée uniquement de r, alors r est le **parcours infixe** de T. Sinon, on suppose que T_1, T_2, ..., T_n sont les sous-arbres de r de gauche à droite dans T. Le **parcours infixe** débute avec le **parcours infixe** de T_1, puis en parcourant r. Il se poursuit par le parcours infixe de T_2, puis le parcours infixe de T_3, et ainsi de suite jusqu'à ce que le parcours infixe de T_n soit effectué.

Exemple de parcours infixe



Parcours infixe: $j\,e\,n\,k\,o\,p\,b\,f\,a\,c\,l\,g\,m\,d\,h\,i$

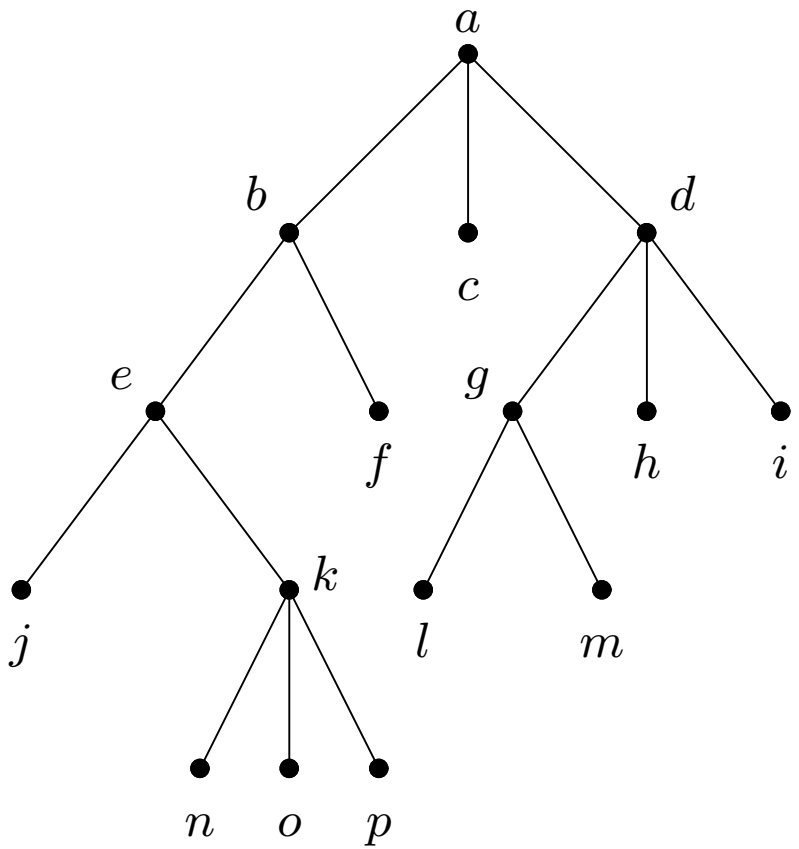
Algorithme: Parcours infixe

procédure *infixe* (T: arborescence ordonnée)
$r :=$ racine de T
si r est une feuille **alors** ranger r dans la liste
sinon
début
    $l :=$ premier fils de r de gauche à droite
    $T(l) :=$ sous-arbre ayant l comme racine
    infixe$(T(l))$
    ranger r dans la liste
    pour chaque fils c de r sauf pour l de gauche à droite
    début
        $T(c) :=$ sous-arbre avec c comme racine
        infixe$(T(c))$
    fin
fin {liste contient le parcours infixe de T}

Définition: parcours postfixe

Soit T une arborescence ordonnée avec la racine r. Si T est constituée uniquement de r, alors r est le **parcours postfixe** de T. Sinon, on suppose que T_1, T_2, ..., T_n sont les sous-arbres de r de gauche à droite dans T. Le **parcours postfixe** débute avec le parcours postfixe de T_1, puis le parcours postfixe de T_2 et ainsi de suite jusqu'au parcours postfixe de T_n, et se termine en parcourant r.

Exemple de parcours postfixe



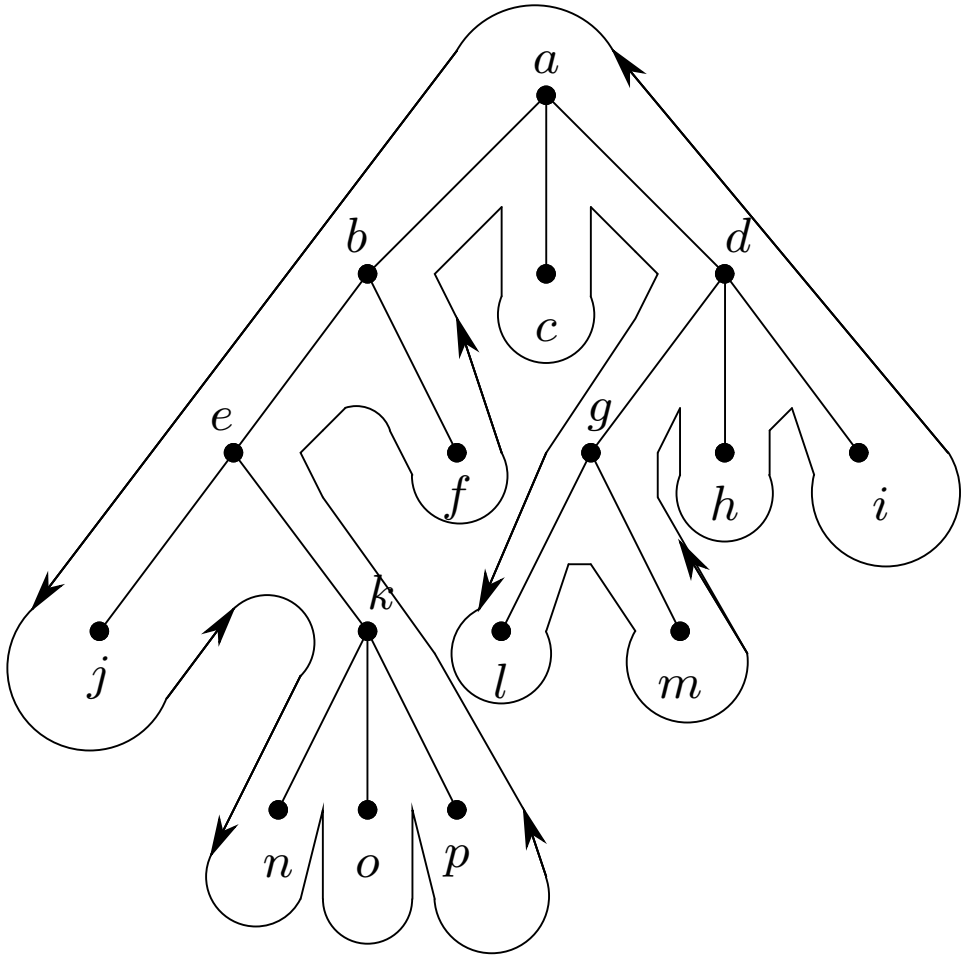
Parcours postfixe: $j\,n\,o\,p\,k\,e\,f\,b\,c\,l\,m\,g\,h\,i\,d\,a$

Algorithme: Parcours postfixe

procédure *postfixe* (T: arborescence ordonnée)
$r :=$ racine de T
pour chaque fils c de r de gauche à droite
début
    $T(c) :=$ sous-arbre avec c comme racine
    postfixe($T(c)$)
fin
ranger r dans la liste
{liste contient le parcours postfixe de T}

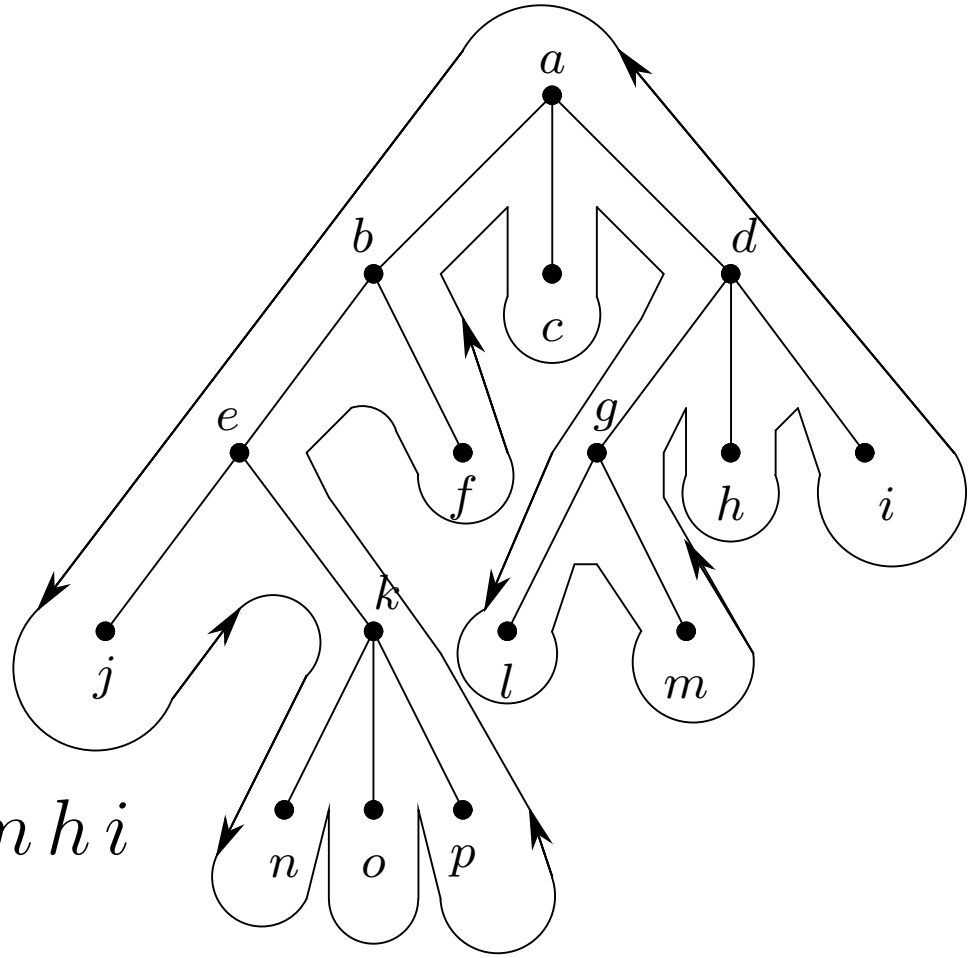
Parcours préfixe, infixe et post-fixe

Imaginons une courbe autour de l'arborescence ordonnée en partant de la racine et en se déplaçant le long des sommets, dans le sens trigonométrique.



Parcours préfixe

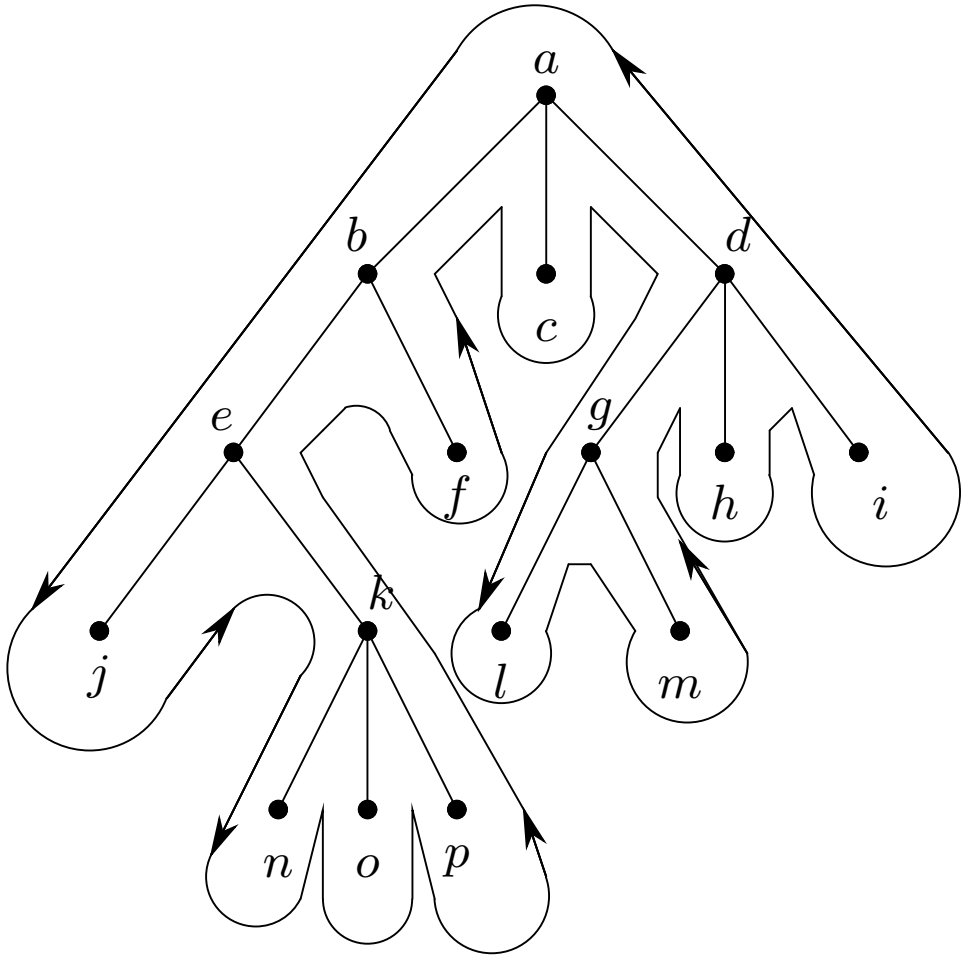
On peut énumérer les sommets en position *préfixe* en énumérant chaque sommet quand cette courbe passe par celui-ci pour la première fois.



$a\,b\,e\,j\,k\,n\,o\,p\,f\,c\,d\,g\,l\,m\,h\,i$

Parcours infixe

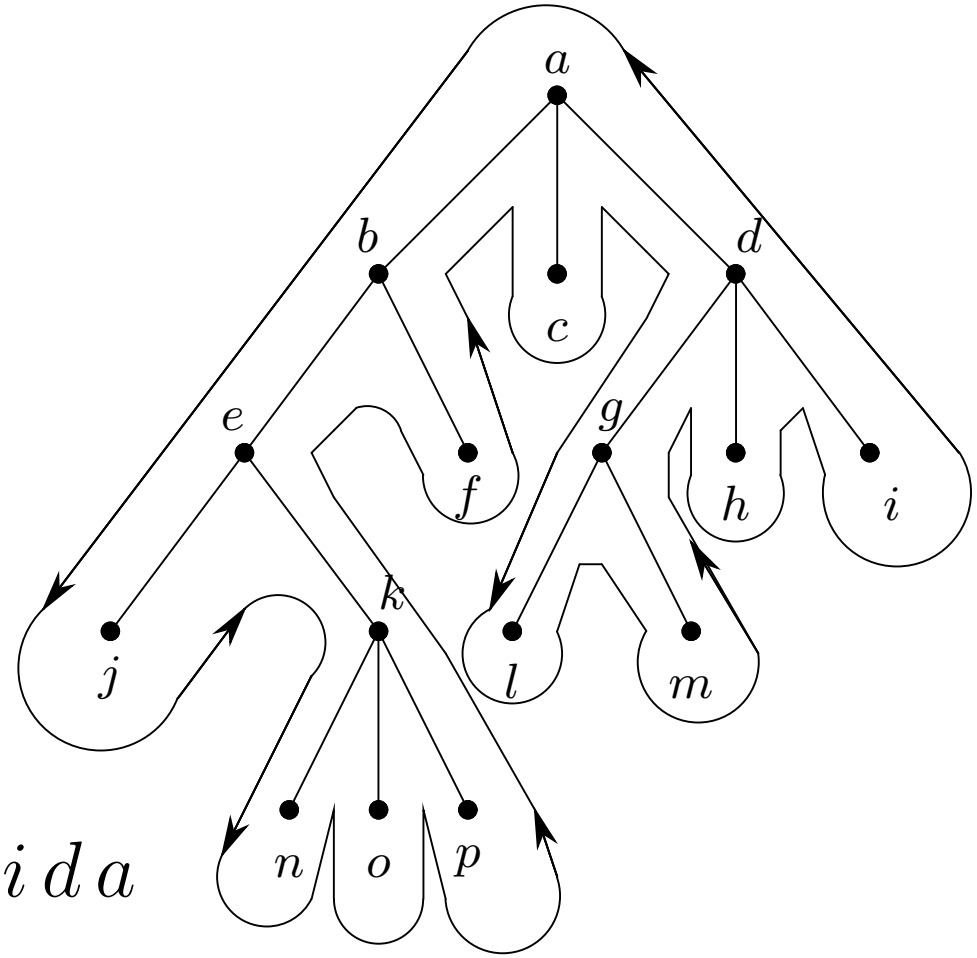
On peut énumérer les sommets en position *infixe* en énumérant une feuille la première fois que cette courbe passe par celle-ci et en énumérant chaque sommet interne la deuxième fois que la courbe passe par celui-ci.



$j\ e\ n\ k\ o\ p\ b\ f\ a\ c\ l\ g\ m\ d\ h\ i$

Parcours postfixe

On peut énumérer les sommets en position *postfixe* en énumérant un sommet la dernière fois que cette courbe passe par celui-ci en revenant vers son père.



$j\ n\ o\ p\ k\ e\ f\ b\ c\ l\ m\ g\ h\ i\ d\ a$

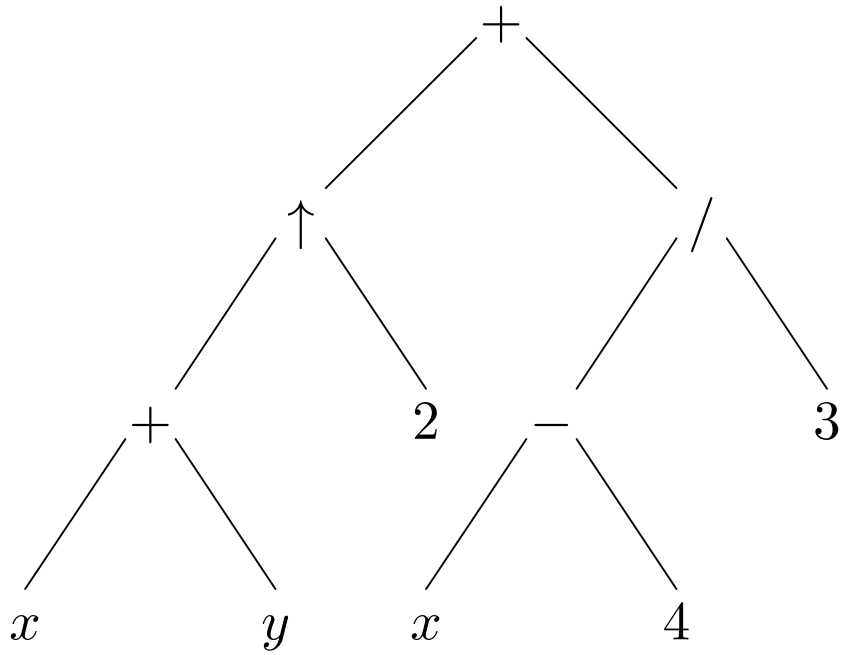
d'expressions arithmétiques et arborescence

Une expression arithmétique comportant les opérateurs $+$, $-$, $*$, $/$ et $\uparrow$ peut se représenter en utilisant une arborescence ordonnée.

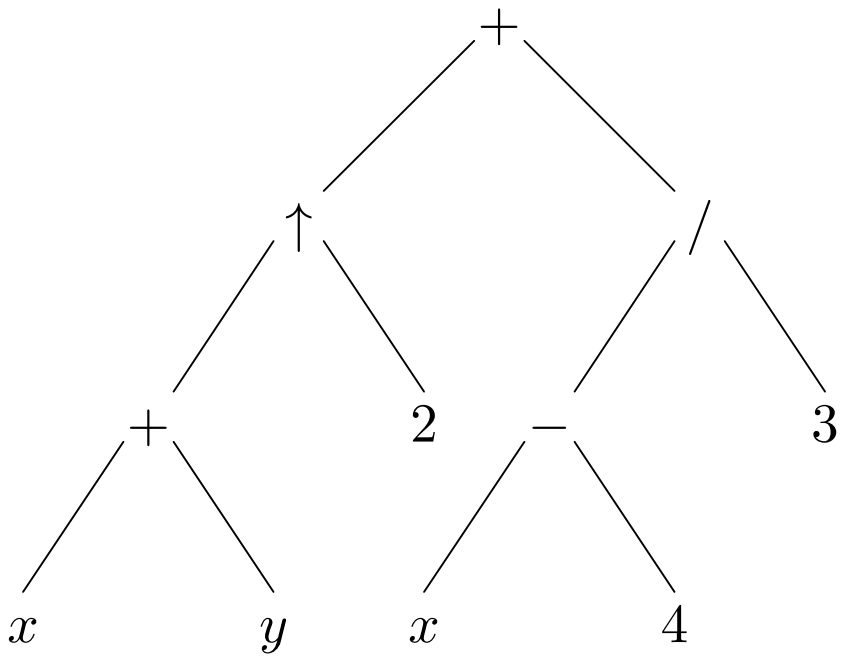
Dans l'arborescence, les sommets internes correspondent aux opérations et les feuilles aux variables ou aux constantes.

Arborescence d'une expression

Arborescence de l'expression
$((x + y) \uparrow 2) + ((x - 4)/3)$:



Parcours infixe d'une expression



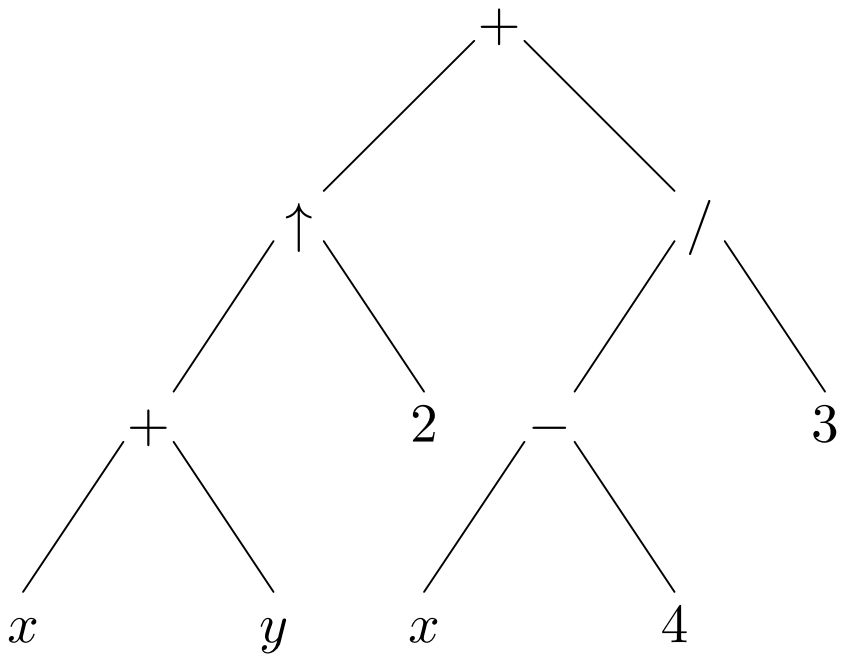
Parcours infixe: $x + y \uparrow 2 + x - 4 / 3$

Forme infixe: $((x + y) \uparrow 2) + ((x - 4)/3))$

Forme infixe d'une expression

Le parcours infixe de l'arborescence préserve l'ordre des termes dans l'expression.
Plusieurs arborescences peuvent avoir le même parcours infixe.
Pour rendre les expressions non ambiguës, il faut inclure les parenthèses dans le parcours infixe lorsqu'on rencontre une opération.
L'expression entièrement mise entre parenthèses est la **forme infixe**.

Parcours préfixe d'une expression



Parcours préfixe: $+ \uparrow + x\, y\, 2\, / \; - \; x\, 4\, 3$

Forme préfixe d’une expression

On obtient la **forme préfixe** d’une expression lorsqu’on parcourt son arborescence de façon préfixe.
Les expressions écrites sous formes préfixe sont en **notation polonaise**.
Une expression en notation préfixée n’est pas ambiguë.

Note historique: Jan Lukasiewicz

Né le 21 décembre 1878 à Lvov.

Mort le 13 février 1956 à Dublin, Irlande.

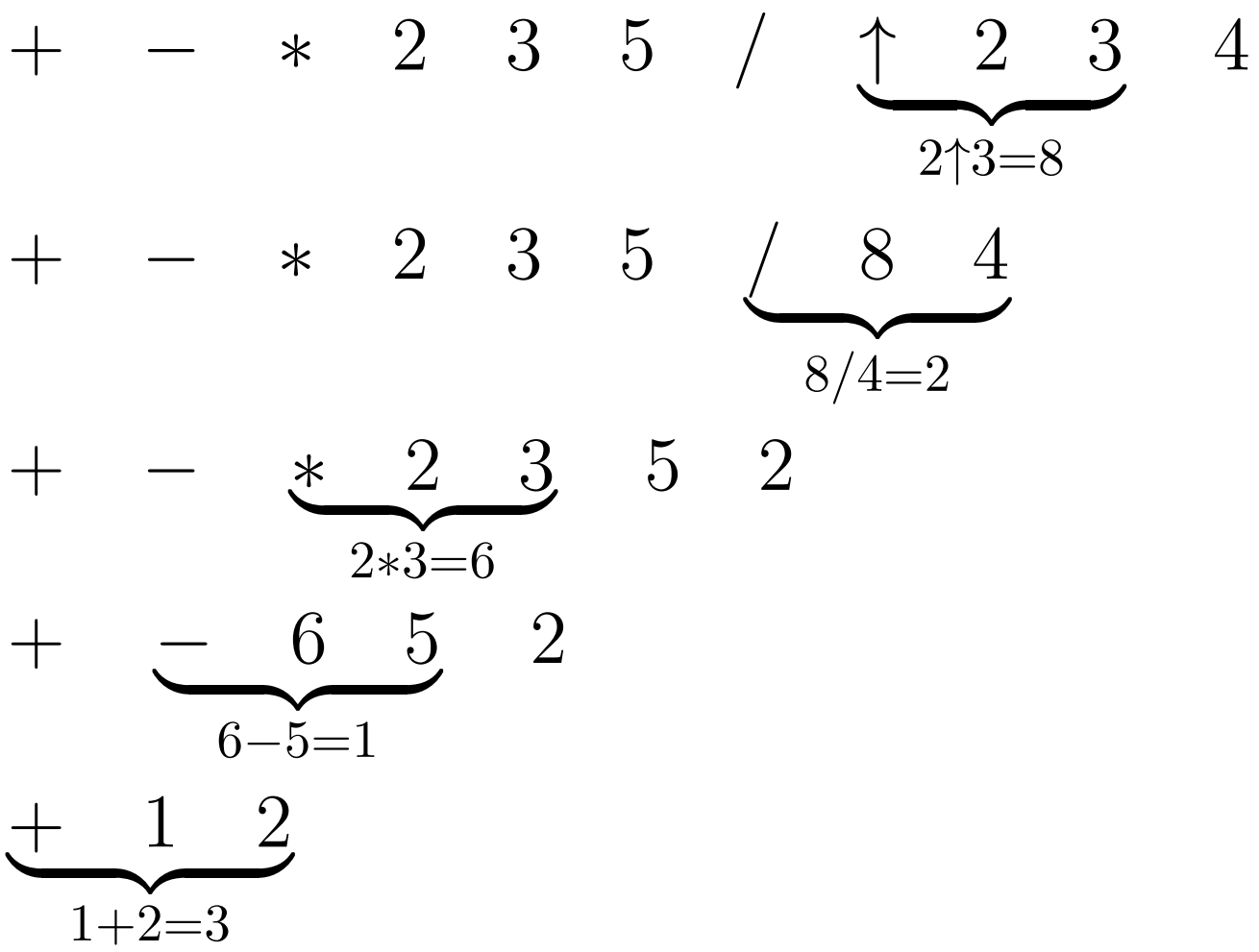
`www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Lukasiewicz.html`

Evaluation d'une expression préfixe

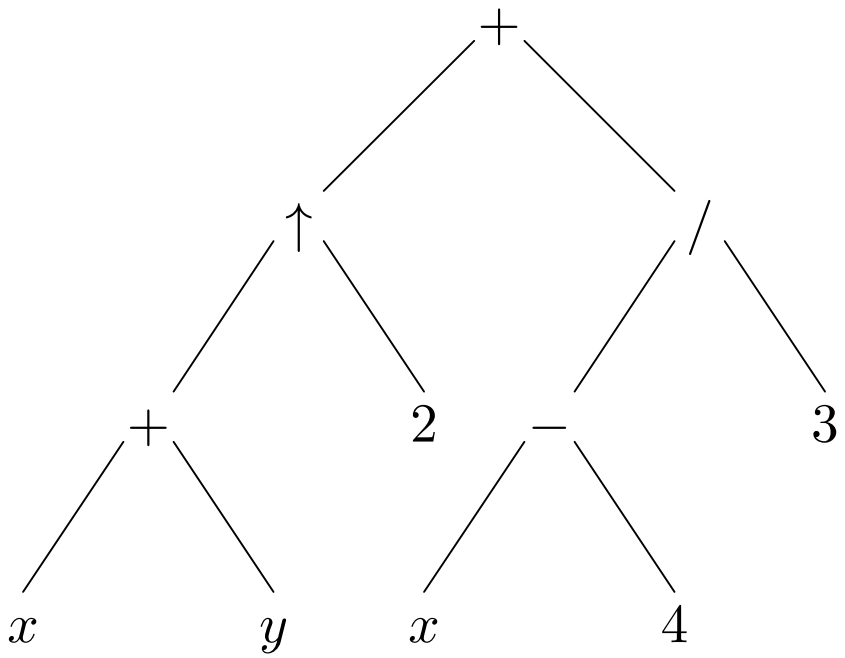
Dans la forme préfixe, les opérateurs binaires précèdent leurs deux opérandes.
On évalue une expression sous forme préfixe de droite à gauche.
Lorsqu'on effectue une opération, on considère le résultat comme un nouvel opérande.

Evaluation d'une expression préfixe

Évaluer l'expression préfixe $+-*235/\uparrow 234$



Parcours postfixe d'une expression



Parcours postfixe: $x\, y\, +\, 2\, \uparrow\, x\, 4\, -\, 3\, /\, +$

Forme et évaluation postfixe d'une expression

On obtient la **forme postfixe** d'une expression lorsqu'on parcourt son arborescence de façon postfixe.
Les expressions écrites sous formes postfixe sont en **notation polonaise inverse**.
Une expression en notation postfixée n'est pas ambiguë.

Dans la forme postfixe, les opérateurs binaires suivent leurs deux opérandes.
On évalue une expression sous forme postfixe de gauche à droite.
Lorsqu'on effectue une opération, on considère le résultat comme un nouvel opérande.

Evaluation d'une expression postfixe

Évaluer l'expression postfixe $723*-4\uparrow93/+$

$$7 \quad \underbrace{2 \quad 3 \quad *}_{2*3=6} \quad - \quad 4 \quad \uparrow \quad 9 \quad 3 \quad / \quad +$$

$$\underbrace{7 \quad 6 \quad -}_{7-6=1} \quad 4 \quad \uparrow \quad 9 \quad 3 \quad / \quad +$$

$$\underbrace{1 \quad 4 \quad \uparrow}_{1\uparrow 4=1} \quad 9 \quad 3 \quad / \quad +$$

$$1 \quad \underbrace{9 \quad 3 \quad /}_{9/3=3} \quad +$$

$$\underbrace{1 \quad 3 \quad +}_{1+3=4}$$

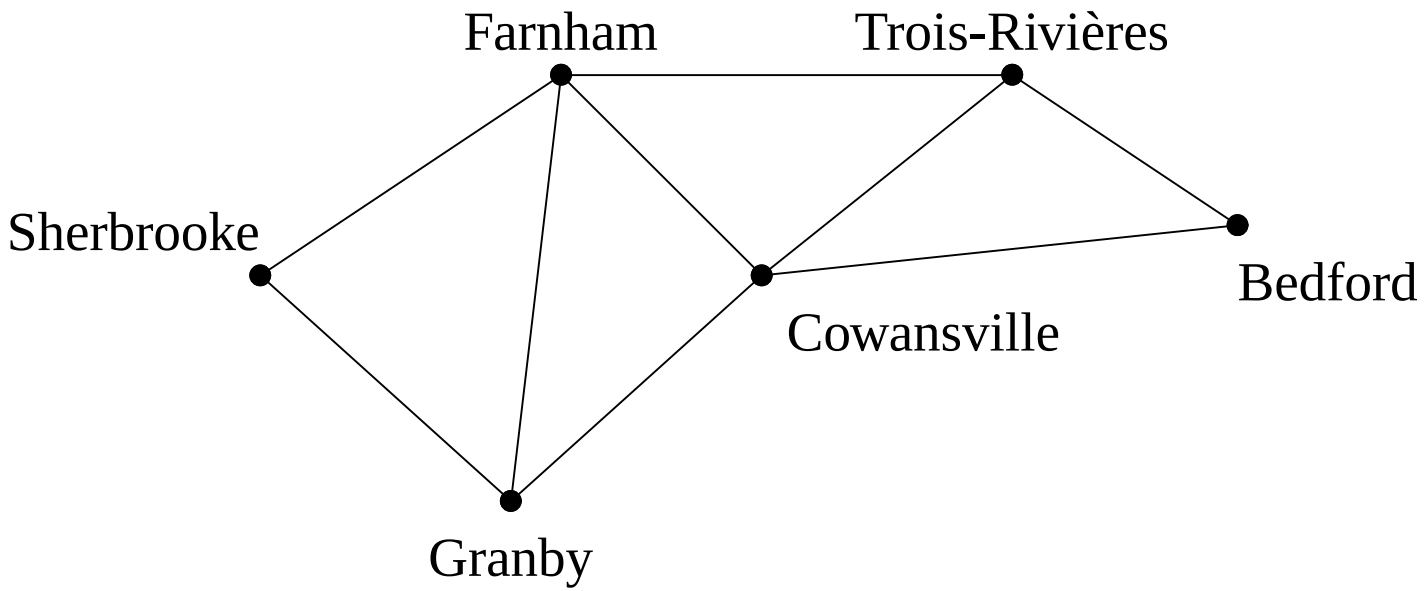
Calculatrice en notation polonaise inverse

perso.wanadoo.fr/noel.jouenne/calc.html

Module Algorithmes sur les arbres

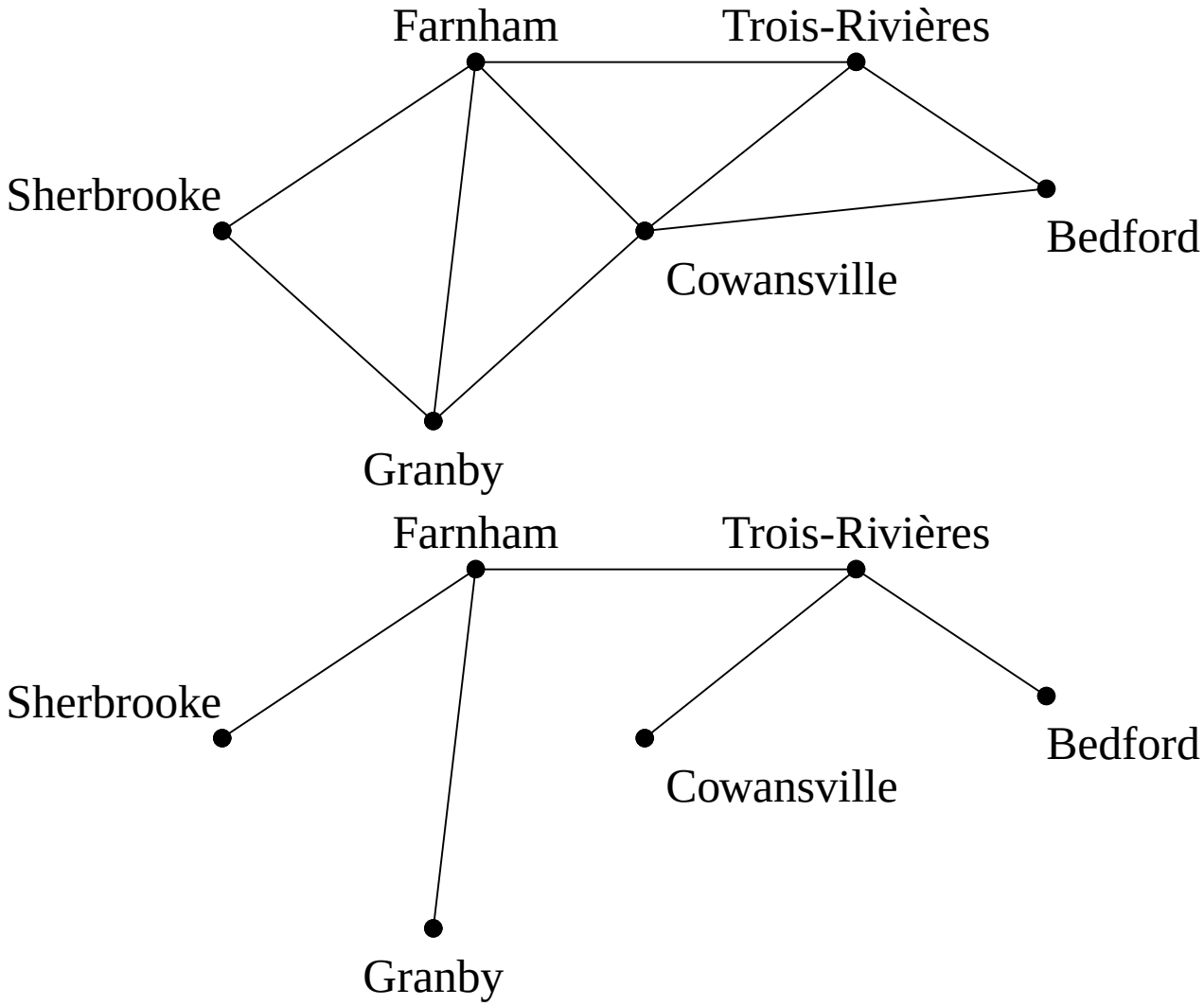
Arbres et algorithmes

Exemple: réseau routier



Le ministère des transports voudrait que chaque ville soit accessible de n'importe quelle autre ville, mais voudrait aussi déneiger le moins grand nombre de routes possibles.

Arbre de recouvrement



Définition: arbre de recouvrement

Soit G un graphe simple. Un **arbre de recouvrement** (ou **couvrant**) de G est un sous-graphe de G qui est un arbre contenant chaque sommet de G.

THÉORÈME: Un graphe simple est connexe si et seulement s'il admet un arbre couvrant.

PREUVE :
$\Leftarrow$) Si $T = (X, F)$ est un arbre couvrant G, la connexité de T entraîne celle de G car $F \subset E$.

Suite page suivante ...

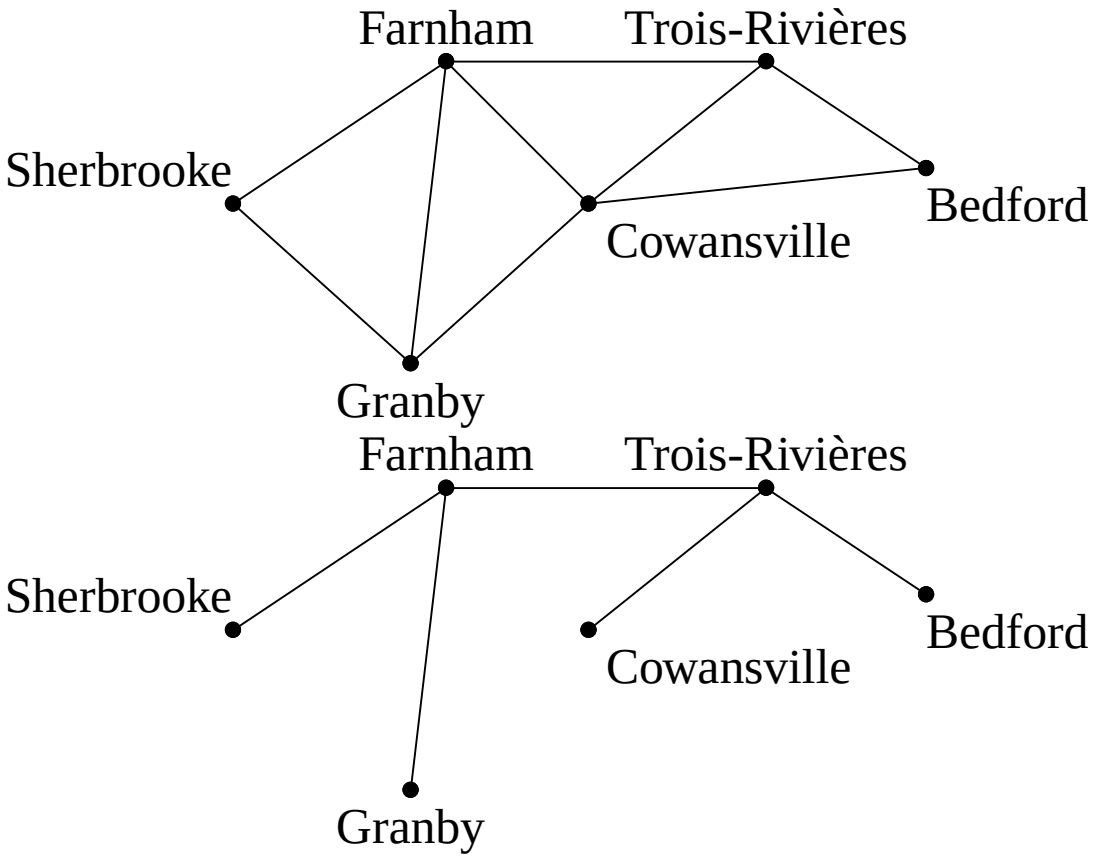
Arbre de recouvrement : suite du théorème

$\Rightarrow$) Réciproquement, si G est connexe, on considère $T = (X, F)$ un graphe partiel de G [a], sans cycle et ayant un nombre maximal d'arcs. On va montrer que T est connexe. Supposons le contraire: il existe deux sommets x et y non reliés (situés dans deux composantes connexes disjointes). On note C_x la composante connexe contenant x. Dans G, il existe une chaîne $\mu = (x_1, ..., x_k)$ avec $x = x_1$ et $y = x_k$. Soit i le plus petit indice tel que x_{i+1} ne soit pas dans C_x. L'arc $(x_i x_{i+1})$ n'est dans aucune composante connexe et $T \cup \{x_i x_{i+1}\}$ est sans cycle (par construction). Cela constitue une contradiction.

[a] c-a-d contenant tous les sommets de G

Algorithme de construction des arbres de recouvrement

On identifie les cycles simples et on enlève des arcs pour “casser” les cycles simples.

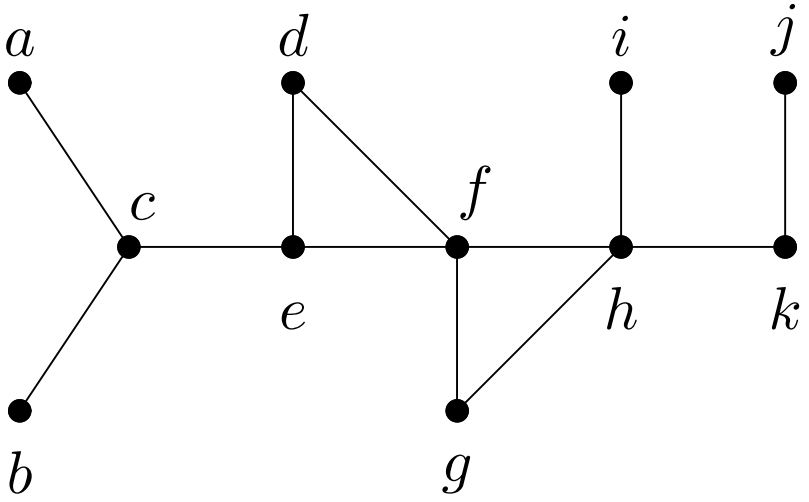


Fouille en profondeur

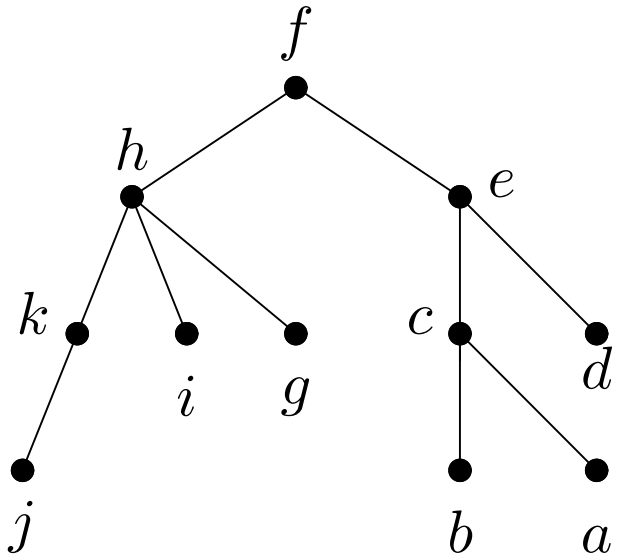
L'idée générale d'une fouille en profondeur d'un graphe consiste à former une arborescence à partir d'un sommet du graphe choisi arbitrairement comme racine. L'arbre de recouvrement constituera le graphe non orienté sous-jacent de cette arborescence.

La fouille en profondeur à partir d'un sommet appelle récursivement la fouille en profondeur des voisins de ce sommet. Notez que si un sommet u a plusieurs voisins v_1, v_2, v_3, ..., la fouille en profondeur de u appelle la fouille en profondeur de v_1, qui elle-même appelle la fouille en profondeur de tous les voisins de v_1, et ainsi de suite récursivement, et ce, avant la fouille en profondeur de v_2, v_3, ...

Exemple de fouille en profondeur



Graphe simple



Arbre de recouvrement

Le sommet f fut choisi comme racine. Le parcours en profondeur d'abord avance tant qu'il peut dans le graphe, et le parcours des chemins laissés de côté se fait lorsqu'on ne peut plus avancer.

Algorithme de fouille en profondeur

procédure *parcourir* (v: un sommet du graphe G)
pour chaque voisin w de v
début
    si w ne se trouve pas dans T **alors**
    début
        mettre les sommet w et l'arc $\{v, w\}$ dans T
        parcourir(w)
    fin
fin

Note: T est l'arborescence en cours de construction.

Suite page suivante...

Algorithme de fouille en profondeur (suite)

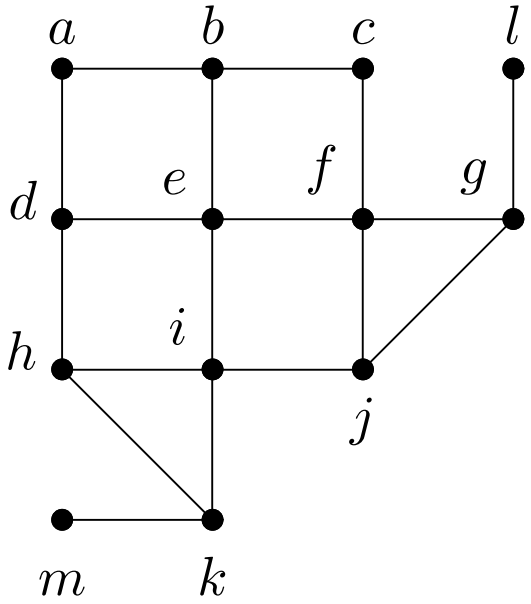
L'algorithme de *fouille en profondeur* consiste en l'appel récursif de *parcourir* avec comme argument un sommet v_i choisi arbitrairement comme racine.

procédure *fouille en profondeur*
(G: graphe simple avec les sommets $v_1, \ldots, v_n$)
$T :=$ arborescence avec v_i comme racine
et aucun autre sommet
parcourir(v_i)
{T est l'arbre souhaité}

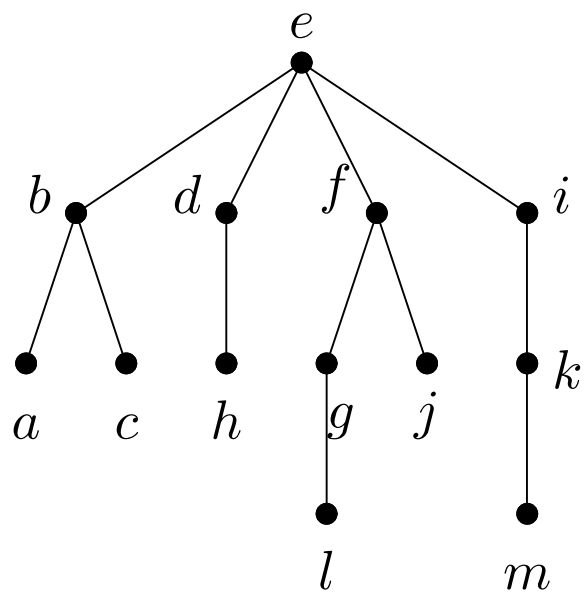
Fouille en largeur

Il faut d'abord choisir arbitrairement un sommet comme racine. L'idée générale consiste à parcourir tous les voisins de la racine, et ensuite à parcourir tous les voisins des voisins, et ensuite à parcourir tous les voisins des voisins des voisins et ainsi de suite.

Exemple de fouille en largeur



Graphe simple



Arbre de recouvrement

Le sommet e est choisi comme racine. Le parcours en largeur parcourt tous les voisins b, d, f et i de e avant de parcourir les voisins a, c, h, g, j et k des voisins b, d, f et i. Et ainsi de suite pour les voisins des voisins des voisins.

Algorithme itératif de fouille en largeur

procédure *fouille en largeur*
(G: graphe simple avec les sommets $v_1, ..., v_n$)
$T :=$ arborescence avec v_i comme racine
sommets traités := liste initialisée avec la racine v_i
sommets atteints := liste initialement vide

Suite page suivante...

Algorithme itératif de fouille en largeur (suite)

tant que *sommets traités* n'est pas vide
début
    pour chaque sommet v de *sommets traités*
    début
        pour chaque voisin w de v
        début
            si w n'est pas dans T **alors**
            début
                mettre w et l'arc $\{v, w\}$ dans T
                ajouter w dans *sommets atteints*
            fin
        fin
    fin
    sommets traités := *sommets atteints*
    sommets atteints := $\emptyset$
fin {T est l'arbre souhaité}

Retour arrière

Un façon de résoudre certains problèmes est de faire une recherche exhaustive des solutions possibles.

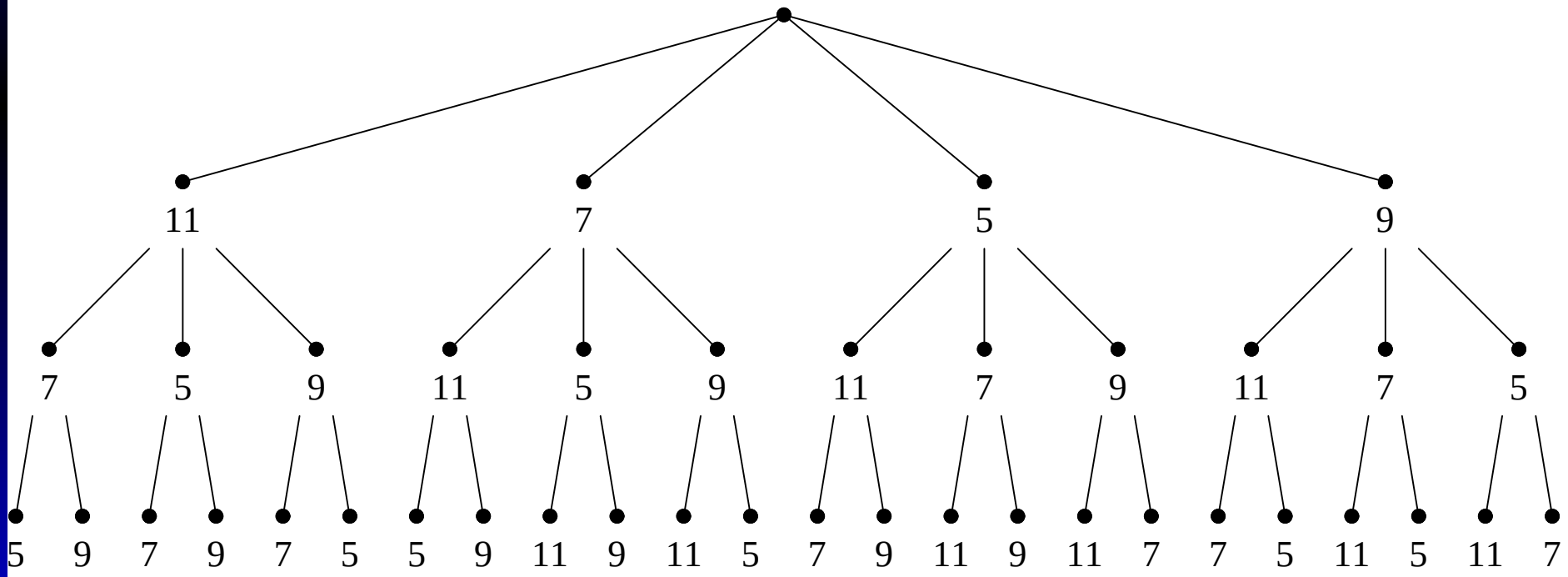
Une façon de chercher systématiquement une solution est d'utiliser un arbre de décision.

Chaque sommet interne représente une décision et chaque feuille une solution possible.

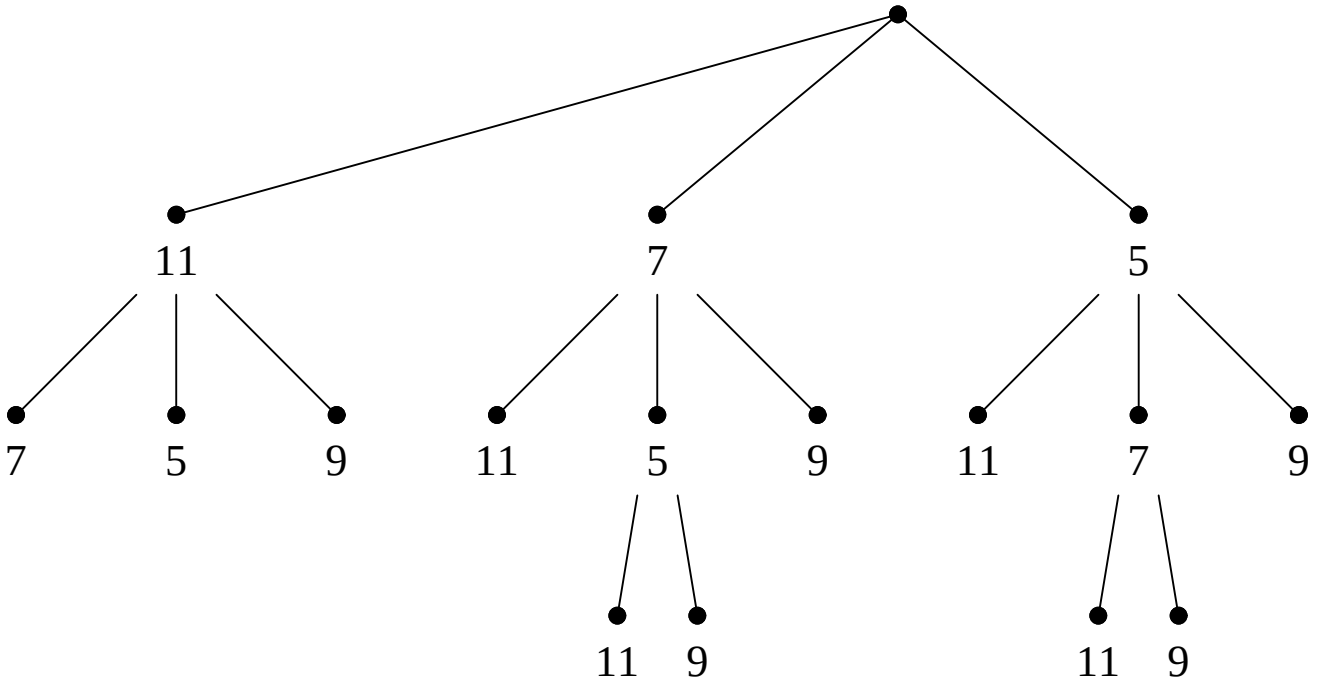
Le problème peut avoir plusieurs solutions. On parcourt l'arbre en profondeur et on arrête la recherche dès qu'on rencontre une solution.

Exemple de retour arrière

PROBLÈME: Rechercher un sous-ensemble propre de $\{11, 7, 5, 9\}$ dont la somme est égale à 14. Voici la liste exhaustive, sous forme d'arbre, de toutes les solutions possibles.

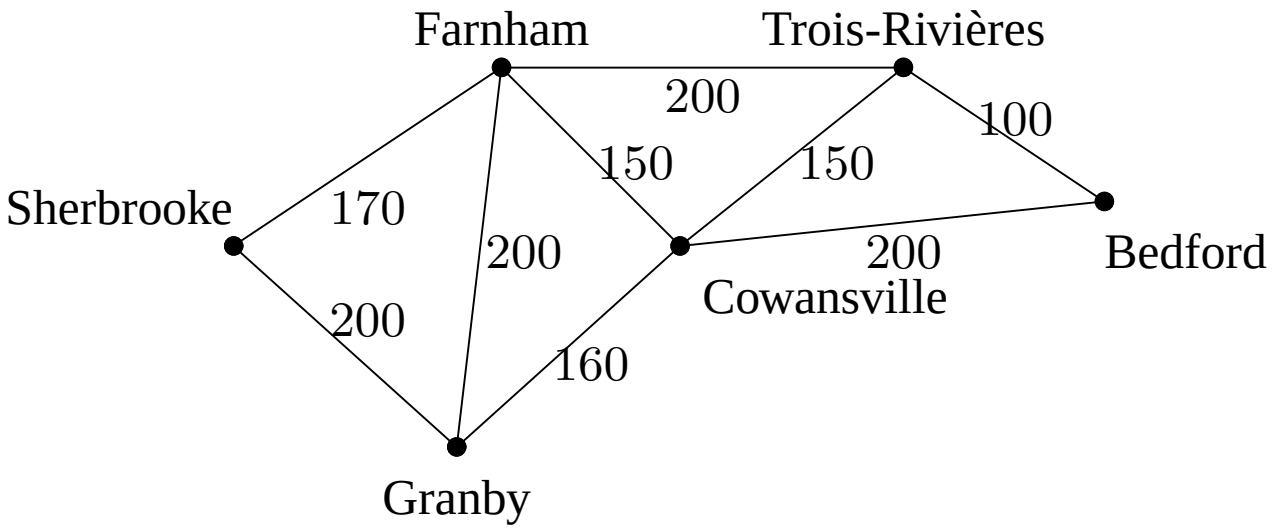


Exemple de retour arrière



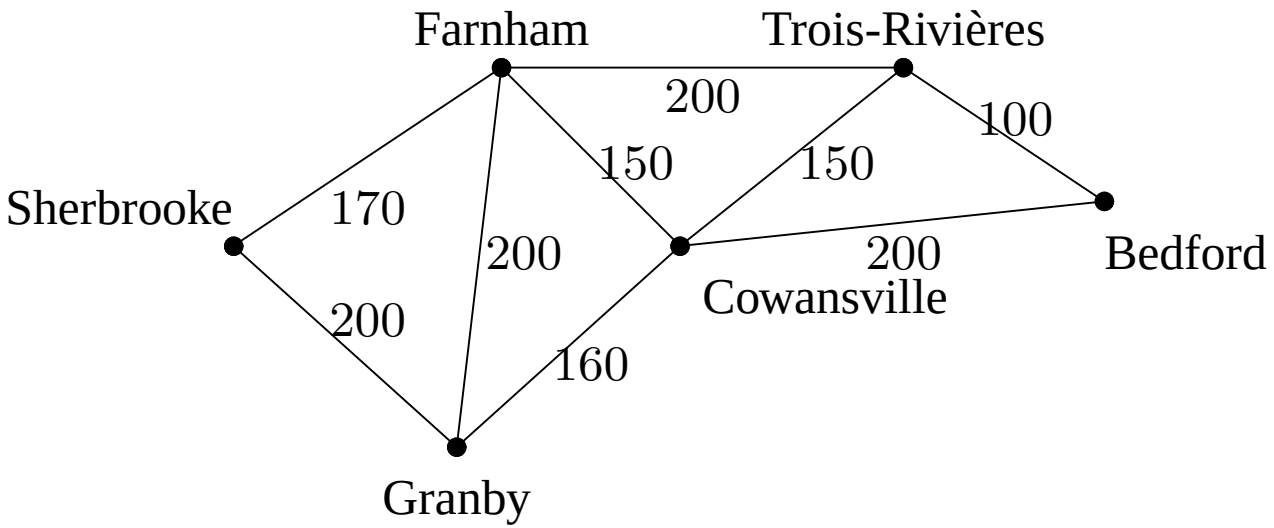
Ce sous-arbre de l'arbre de toutes les solutions possibles représente le parcours en profondeur effectué jusqu'à ce qu'on rencontre la première solution $\{5, 9\}$.

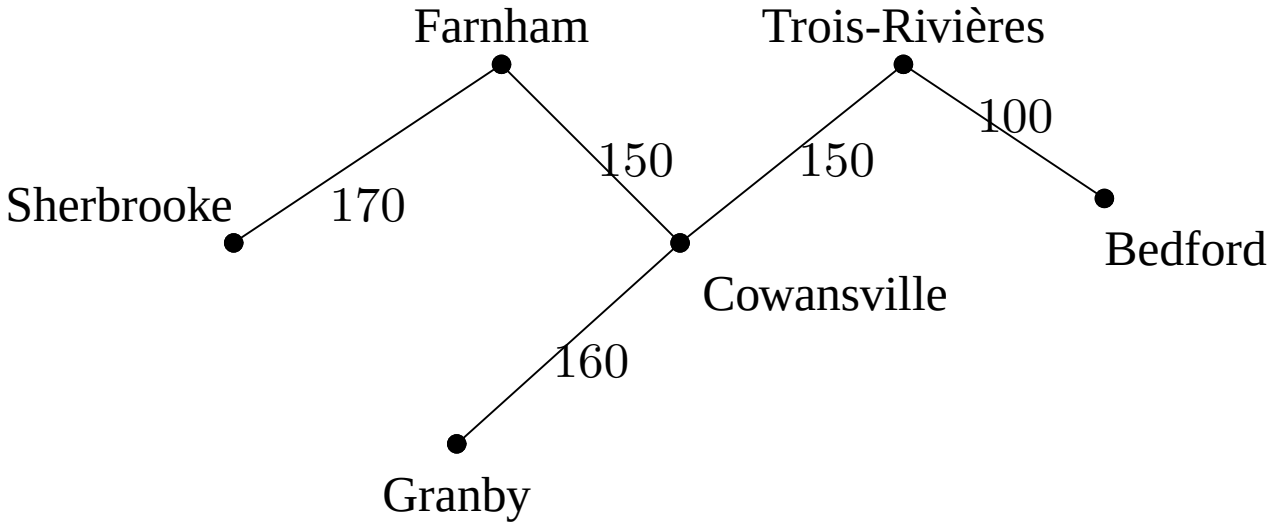
Réseau routier de coût minimal



Le ministère des transports voudrait que chaque ville soit accessible de n'importe quelle autre ville, mais voudrait aussi déneiger le moins grand nombre de *kilomètres* possibles.

Réseau routier de coût minimal





Définition: arbre de recouvrement minimal

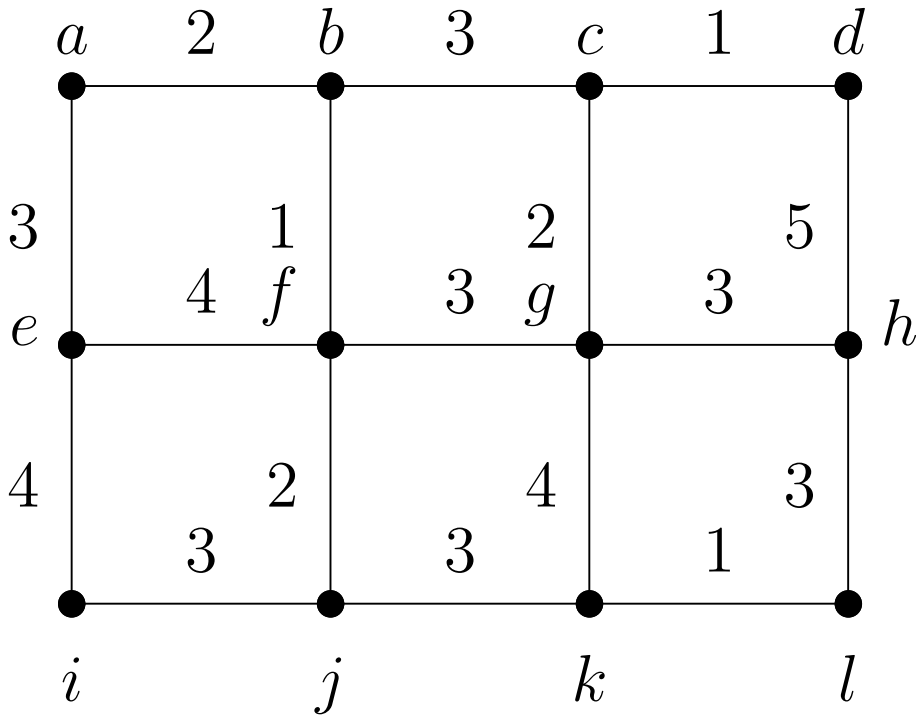
Un **arbre de recouvrement minimal** dans un graphe valué connexe est un arbre de recouvrement dont la somme des coûts est minimale.

Algorithme de Prim
Robert C. Prim, *Shortest connection networks and some generalizations*, Bell System Techn. J. vol 36 (1957) pp. 1389–1401.

Algorithme de Prim

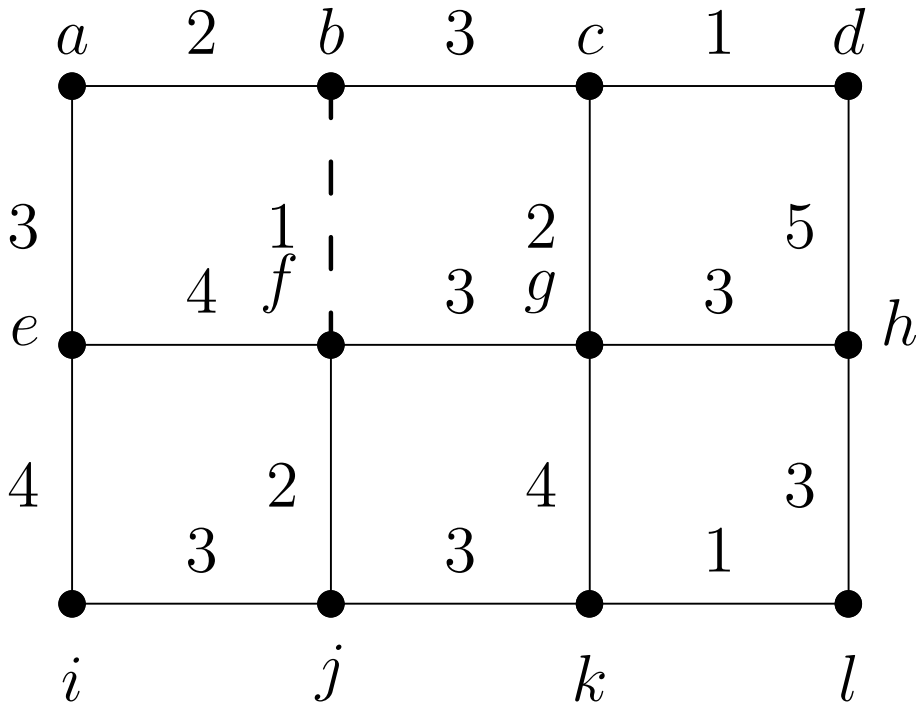
procédure *Prim* (G: graphe non orienté connexe valué
à n sommets)
$T :=$ un arc de coût minimal
pour $i := 1$ **à** $n - 2$
début
$e :=$ un arc de coût minimal incident à un sommet
dans T et ne formant pas un cycle simple
dans T si il est ajouté à T.
$T := T$ avec l'arc e ajouté
fin
$\{T$ est un arbre de recouvrement minimal de $G\}$

Exemple de l'algorithme de Prim, étape 1 de 5



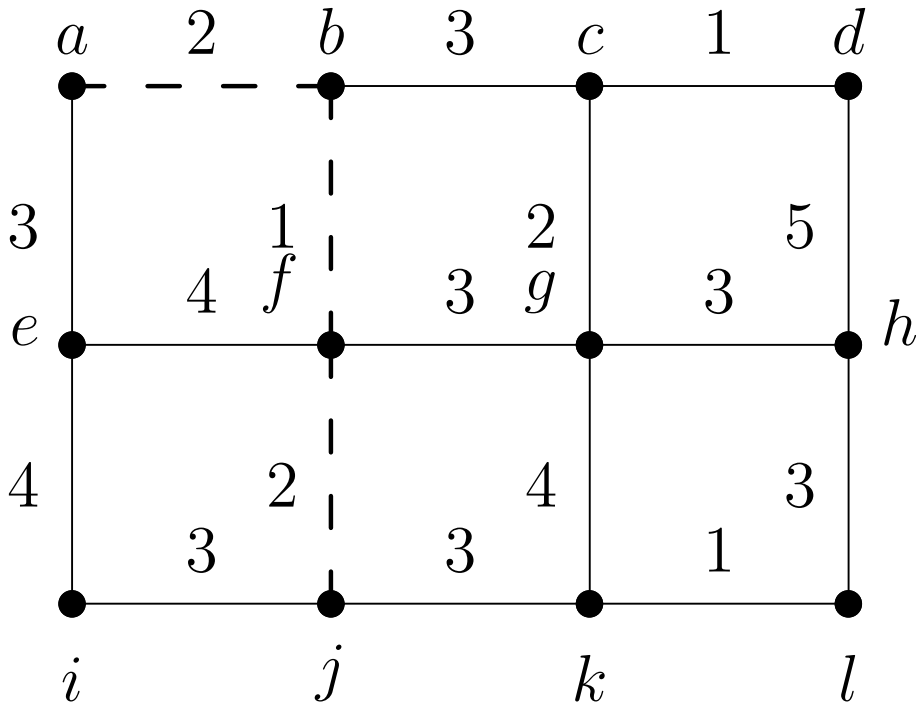
Trouve les arcs $\{b, f\}$, $\{c, d\}$ et $\{k, l\}$ de coût minimal.

Exemple de l'algorithme de Prim, étape 2 de 5



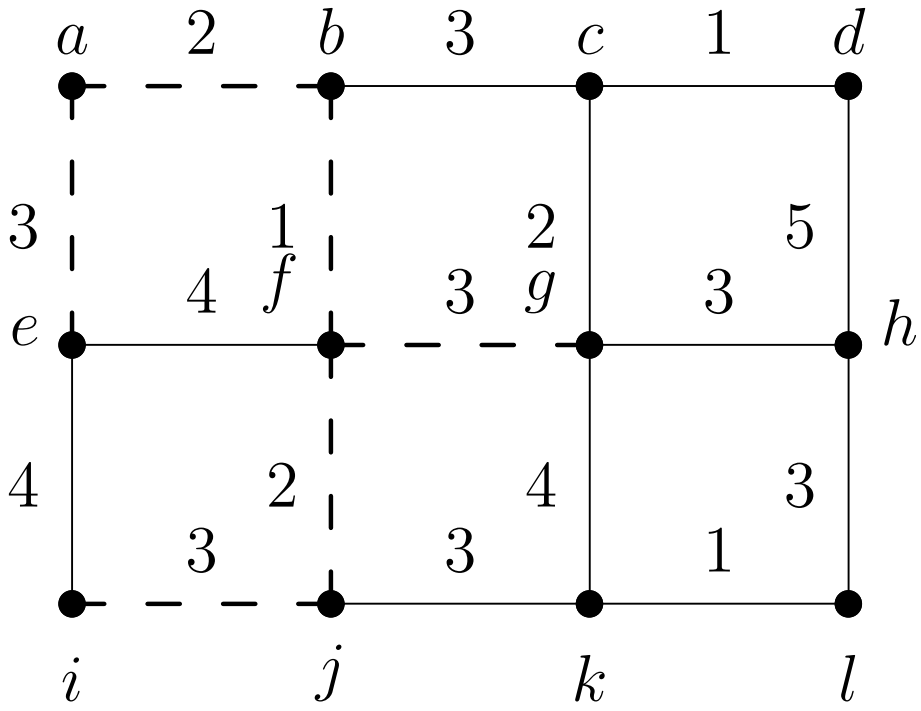
Initialise T avec l'arc $\{b, f\}$ choisi arbitrairement parmi les arcs de coût minimal.

Exemple de l'algorithme de Prim, étape 3 de 5



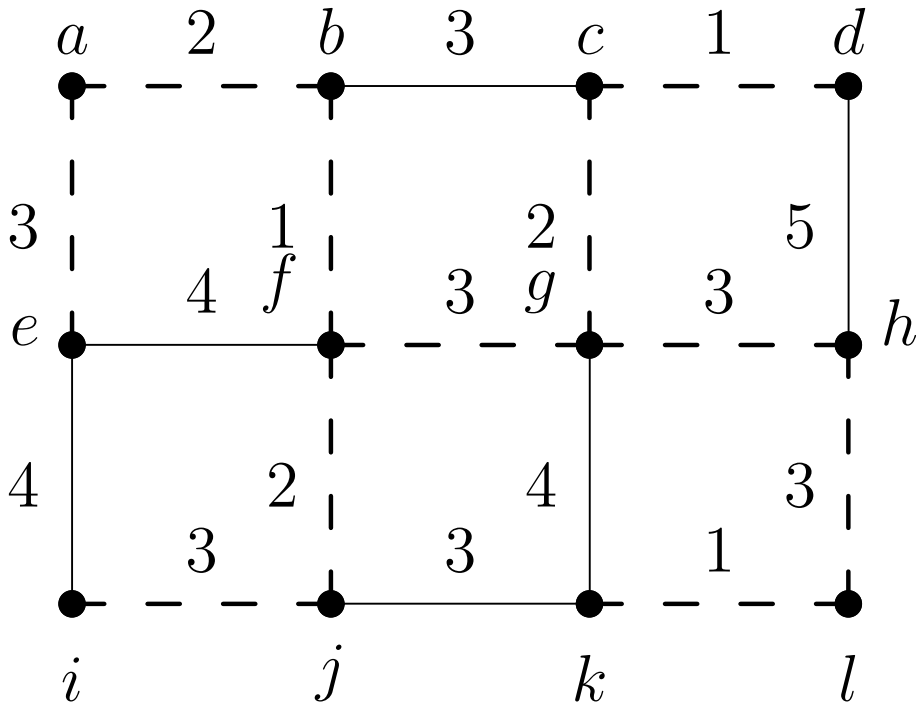
Ajout des arcs $\{a, b\}$ et $\{f, j\}$ à T.

Exemple de l'algorithme de Prim, étape 4 de 5



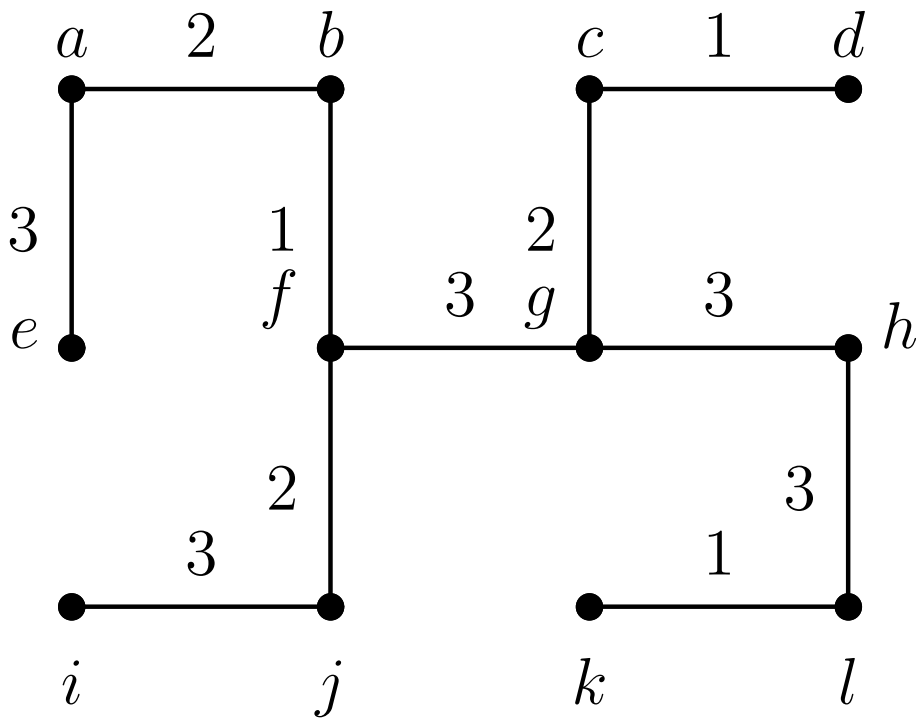
Ajout des arcs $\{a, e\}$, $\{i, j\}$ et $\{f, g\}$ à T

Exemple de l'algorithme de Prim, étape 5 de 5



Ajout des arcs $\{c, g\}$, $\{c, d\}$, $\{g, h\}$, $\{h, l\}$ et $\{k, l\}$ à T.

Exemple de l'algorithme de Prim



Solution finale.

Analyse de l'algorithme de Prim

THÉORÈME : L'algorithme de Prim calcule un arbre recouvrant de poids minimum.

PREUVE : En exercice

Complexité de l'algorithme de Prim

Pour chaque sommet x de $X \backslash T$, il faut connaître l'arc f de poids minimal qui le relie aux autres sommets, et son poids $poids_A(x) = poids(f)$.
Chaque étape de sélection coûte $O(n)$. Le traitement coûte $O(d^+(T))$.
Au total on a donc $O(n^2)$ pour la sélection et $\sum_{x \in X} O(d^+(x))$ c'est-à dire $O(n+m)$.
Finalement, la complexité de l'algorithme est $O(n^2)$.

Note historique: Joseph B. Kruskal

Room 2C-281, Bell Laboratories, Lucent Technologies, 700 Mountain Av. Murray Hill, NJ 07974-0636,

Algorithme de Kruskal

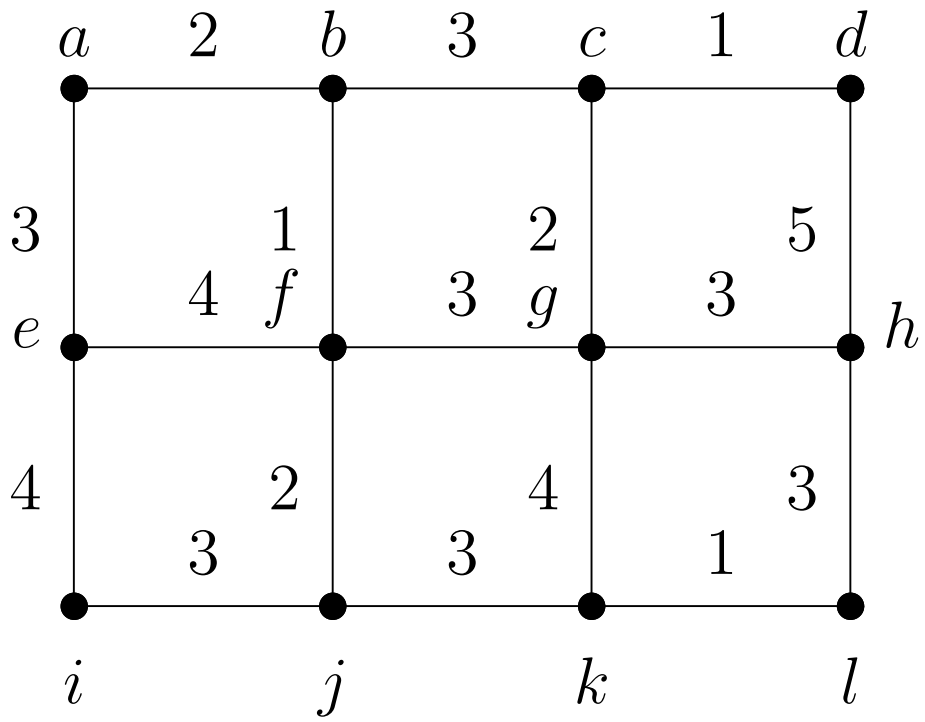
procédure *Kruskal* (G: graphe non orienté connexe valué
à n sommets)
$T :=$ graphe vide
liste $:=$ liste triée des arcs de G
tant que T contient moins de $n - 1$ arcs
début
$e :=$ l'arc suivant de la *liste*
si e ne forme pas un cycle simple
lorsqu'il est ajouté à T,
alors $T := T$ avec l'arc e ajouté.
fin
{T est un arbre de recouvrement minimal de G}

Algorithme de Kruskal

À noter la différence entre l'algorithme de Prim et de Kruskal. Dans l'algorithme de Prim, on choisit les arcs de coût minimal qui sont incidents à un sommet se trouvant déjà dans l'arbre en cours de construction et ne formant pas un cycle.

Dans l'algorithme de Kruskal, il faut d'abord trier les arcs en ordre croissant de coût. Contrairement à l'algorithme de Prim, l'algorithme de Kruskal choisit des arcs de coût minimal qui ne sont pas nécessairement incidents à un sommet déjà dans l'arbre en cours de construction et ne formant pas un cycle.

Exemple de l'algorithme de Kruskal, étape 1 de 4

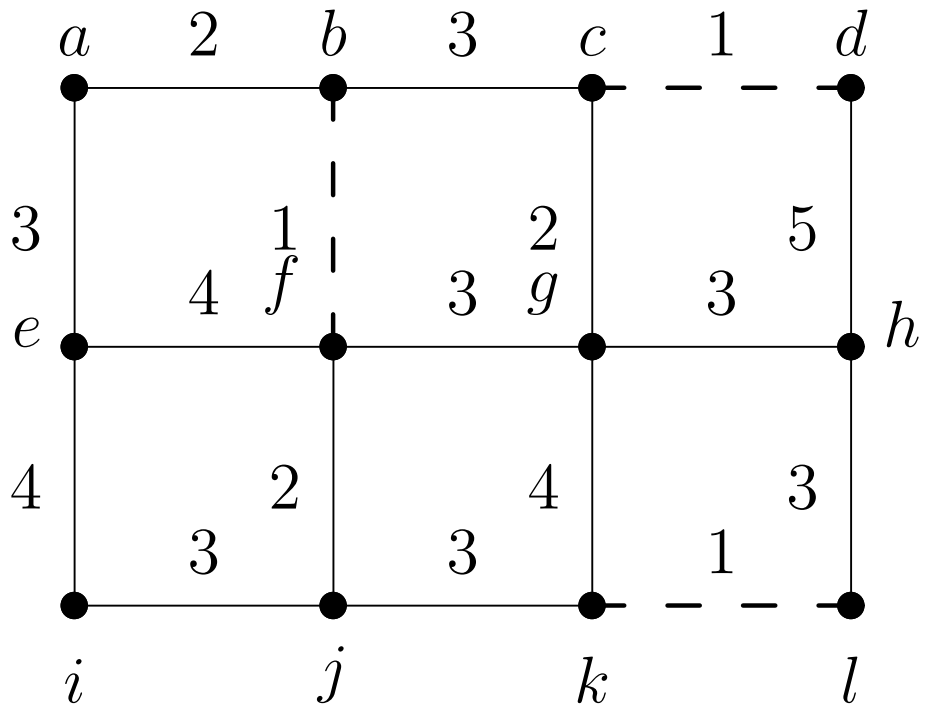


Tri des arcs en ordre croissant de coût.

arc	$
$\{g, h\}$	3
$\{h, l\}$	3

arc	$
$\{b, f\}$	1
$\{c, d\}$	1
$\{k, l\}$	1
$\{a, b\}$	2
$\{c, g\}$	2
$\{f, j\}$	2
$\{a, e\}$	3
$\{b, c\}$	3
$\{f, g\}$	3

Exemple de l'algorithme de Kruskal, étape 2 de 4

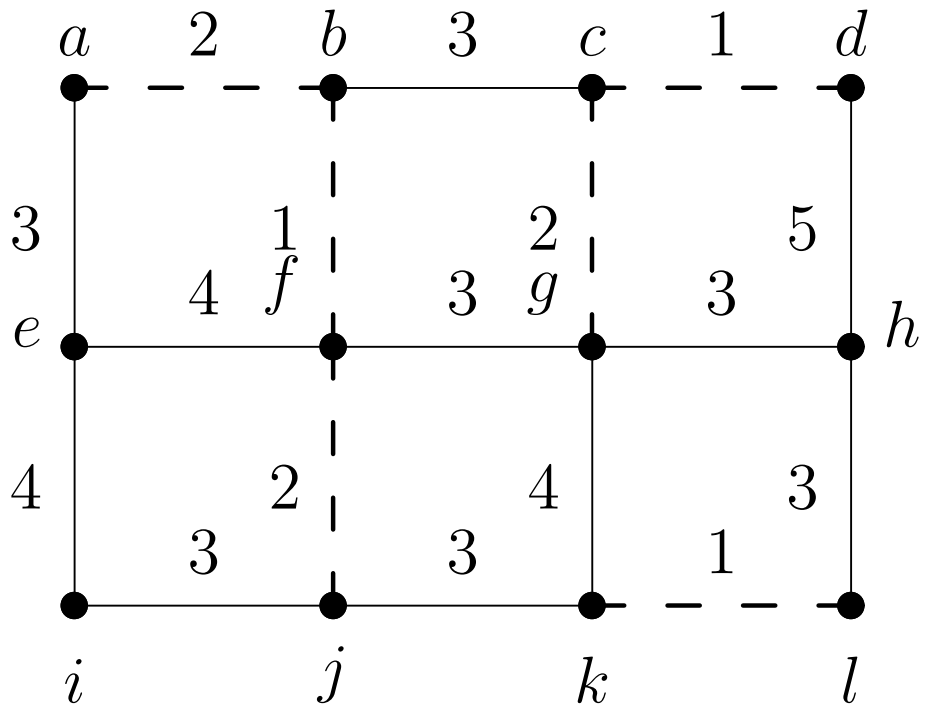


Ajout des arcs $\{b, f\}$, $\{c, d\}$ et $\{k, l\}$.

arc	$
$\{g, h\}$	3
$\{h, l\}$	3

arc	$
$\{b, f\}$	**1**
$\{c, d\}$	**1**
$\{k, l\}$	**1**
$\{a, b\}$	2
$\{c, g\}$	2
$\{f, j\}$	2
$\{a, e\}$	3
$\{b, c\}$	3
$\{f, g\}$	3

Exemple de l'algorithme de Kruskal, étape 3 de 4

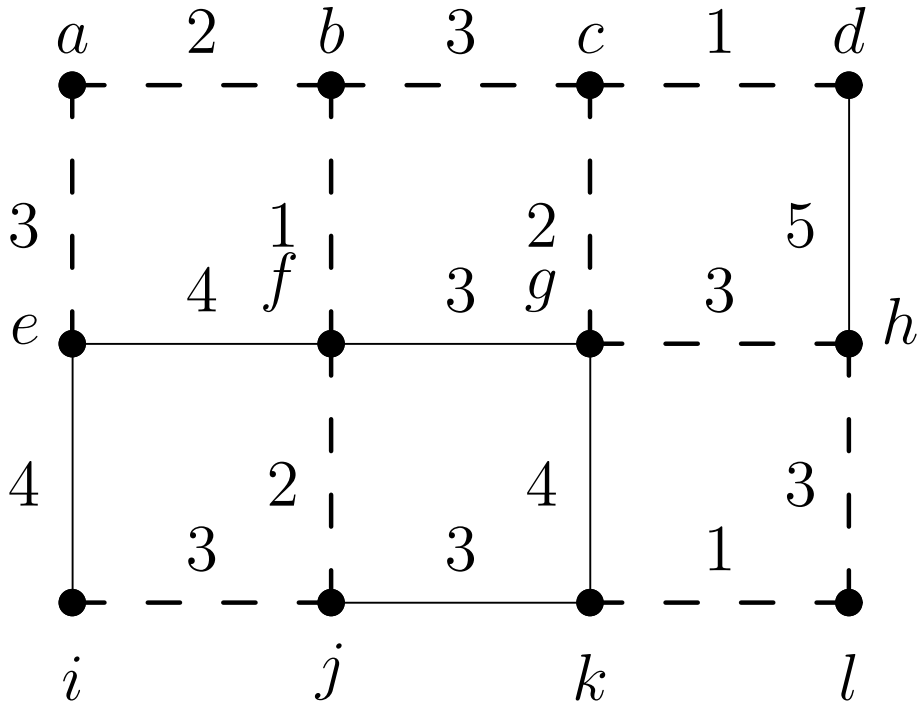


Ajout des arcs $\{a, b\}$, $\{c, g\}$ et $\{f, j\}$.

arc	$
$\{g, h\}$	3
$\{h, l\}$	3

arc	$
$\{b, f\}$	1
$\{c, d\}$	1
$\{k, l\}$	1
$\{a, b\}$	**2**
$\{c, g\}$	**2**
$\{f, j\}$	**2**
$\{a, e\}$	3
$\{b, c\}$	3
$\{f, g\}$	3

Exemple de l'algorithme de Kruskal, étape 4 de 4



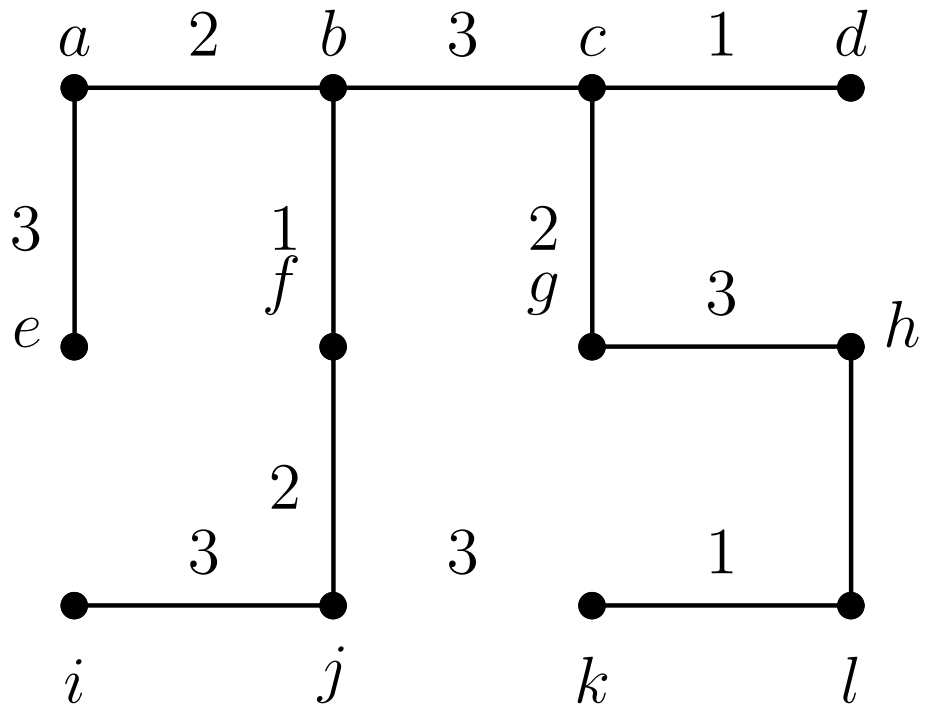
arc	$
$\{b, f\}$	1
$\{c, d\}$	1
$\{k, l\}$	1
$\{a, b\}$	2
$\{c, g\}$	2
$\{f, j\}$	2
$\{a, e\}$	**3**
$\{b, c\}$	**3**
$\{f, g\}$	3

Ajout des arcs $\{a, e\}$, $\{b, c\}$, $\{g, h\}$, $\{h, l\}$ et $\{i, j\}$. L'arc $\{f, g\}$ n'a pas été ajouté car il formait un cycle simple.

Exemple de l'algorithme de Kruskal, étape 4 de 4 (suite)

arc	$
$\{g, h\}$	**3**
$\{h, l\}$	**3**
$\{i, j\}$	**3**
$\{j, k\}$	3
$\{e, f\}$	4
$\{e, i\}$	4
$\{g, k\}$	4
$\{d, h\}$	5

Exemple de l'algorithme de Kruskal



Solution finale.

arc	$
$\{b, f\}$	**1**
$\{c, d\}$	**1**
$\{k, l\}$	**1**
$\{a, b\}$	**2**
$\{c, g\}$	**2**
$\{f, j\}$	**2**
$\{a, e\}$	**3**
$\{b, c\}$	**3**
$\{f, g\}$	3

Exemple de l'algorithme de Kruskal (suite)

arc	$
$\{g, h\}$	**3**
$\{h, l\}$	**3**
$\{i, j\}$	**3**
$\{j, k\}$	3
$\{e, f\}$	4
$\{e, i\}$	4
$\{g, k\}$	4
$\{d, h\}$	5

Analyse de l'algorithme de Kruskal

THÉORÈME : Si G est un graphe connexe, l'algorithme de Kruskal construit un arbre couvrant de poids minimum.

COMPLEXITÉ : Soient $n = |X|$ et $m = |E|$.
Le tri des arêtes se fait en $O(mlogm) = O(mlogn)$. L'initialisation se fait en temps constant.
La boucle 'tant que' est exécutée au pire m fois. Pour tester si $T \cup \{e\}$ contient un cycle (avec $e = xy$), on fait un parcours en largeur dans T à partir de x. Si on rencontre y, $T \cup \{e\}$ contient un cycle. Sinon, il n'en contient pas. Le coût du parcours est $O(n)$.
La complexité totale est donc $O(nm)$.

Module Dénombrements

Dénombrement

Le modèle de l'urne

Il y a n objets dans une urne. On va choisir séquentiellement $0 \leq r \leq n$ objets dans cette urne. Combien de séquences différentes de r objets est-il possible d'obtenir?
Il y a deux critères à considérer:

- La **remise**
  - Après avoir choisi un objet dans l'urne, on le remet dedans (**sélection avec remise**).
  - Après avoir choisi un objet dans l'urne, on ne le remet pas dedans (**sélection sans remise**).
- L'**ordre**
  - L'ordre de sélection des objets importe.
  - L'ordre de sélection des objets est sans importance.

Définition: permutation

La **permutation** d'un ensemble d'objets distincts est un arrangement *ordonné* de ces objets. Si E est un

ensemble non vide , une **permutation** des éléments de E est une *bijection* $f : E \to E$. Un arrangement

ordonné de r éléments d'un ensemble est appelé une r**-permutation**.

Le nombre de r-permutations d'un ensemble de n éléments est noté $P(n, r)$.

Dans le modèle de l'urne, une permutation est une séquence ordonnée sans remise.

Théorème: Le nombre de r-permutations

THÉORÈME: Le nombre de r-permutations d'un ensemble de n éléments distincts est

$$P(n,r) = n(n-1)(n-2)\cdots(n-r+1) = \frac{n!}{(n-r)!}.$$

Exemple : Dans un tiercé sur 15 chevaux, il y a $P(15,3) = 15 \times 14 \times 13 = 2730$ possibilités. Si c'est un quarté, il y en a : $P(15,4) = 15 \times 14 \times 13 \times 12 = 32760$.

Note:
$P(n,n) = n!$
$P(n,0) = 1$

Définition: combinaison

Une **combinaison** d'un ensemble d'objets distincts est un arrangement *non ordonné* de ces objets.

Une r-**combinaison** des éléments d'un ensemble est une sélection non ordonnée de r éléments de l'ensemble. Ainsi, une r-combinaison constitue simplement un sous-ensemble de l'ensemble ayant r éléments.

Le nombre de r-combinaisons d'un ensemble de n éléments distincts est noté $C(n, r)$.

Définition: combinaison (suite)

Dans le modèle de l'urne, une combinaison est une séquence non ordonnée sans remise.

La différence entre un arrangement et une combinaison est que les éléments de l'arrangement sont numérotés, alors que ceux de la combinaison sont en vrac.

Théorème: Le nombre de r-combinaisons

THÉORÈME: Le nombre de r-combinaisons d'un ensemble de n éléments, où n est un entier positif et r, un entier avec $0 \leq r \leq n$, est égal à

$$C(n,r) = \frac{n!}{r!(n-r)!}.$$

DÉMONSTRATION: Le nombre de r-permutations d'un ensemble peut être obtenu en formant $C(n,r)$ r-combinaisons de l'ensemble, puis en ordonnant les éléments dans chaque r-combinaison ce qui peut être fait de $P(r,r)$ manières. Par conséquent $P(n,r) = C(n,r).P(r,r)$ Ce qui implique

$$C(n,r) = \frac{P(n,r)}{P(r,r)} = \frac{n!/(n-r)!}{r!/(r-r)!} = \frac{n!}{r!(n-r)!}$$

Définition: coefficient binomial

Note: $C(n, n) = 1$;
$C(n, 0) = 1$;
$C(n, r) = C(n, n - r)$ avec $r \leq n$.

Il existe une autre notation courante pour le nombre de r-combinaisons d'un ensemble à n éléments, notamment

$$C(n, r) = \binom{n}{r}.$$

Ce nombre s'appelle aussi **coefficient binomial**.

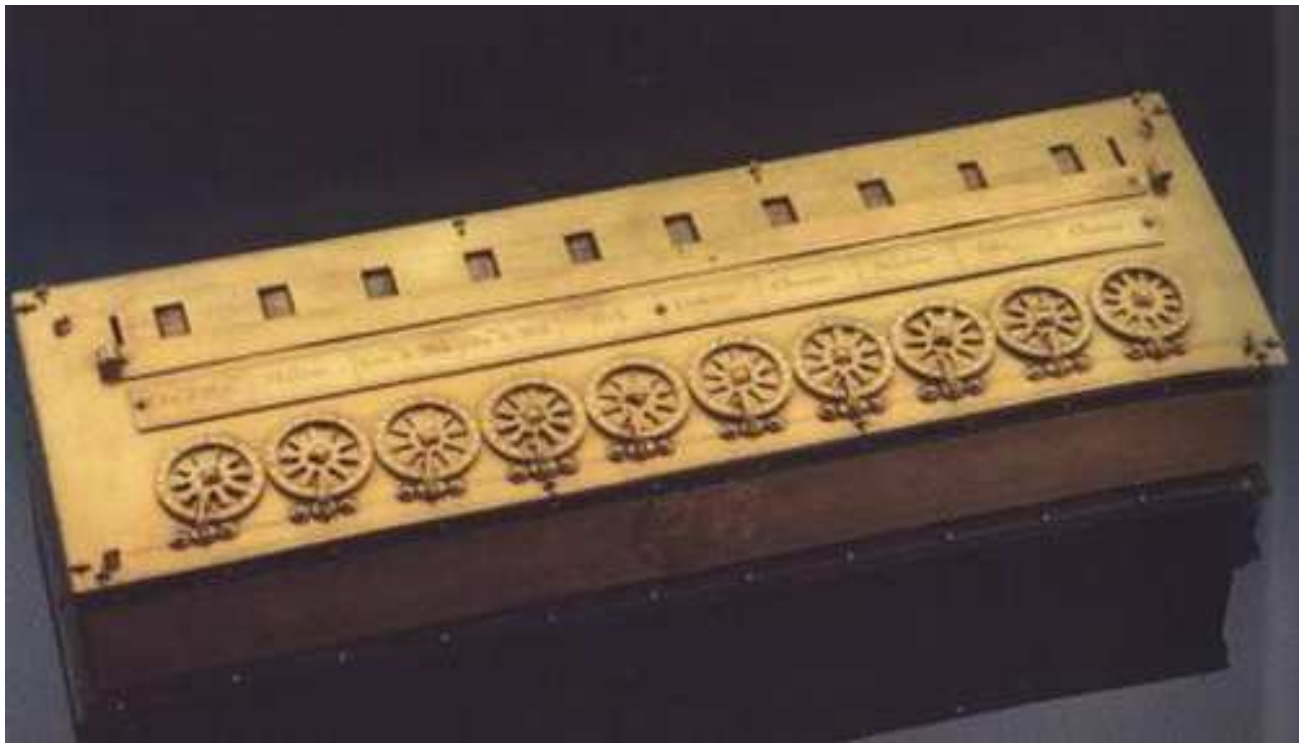
Note historique: Blaise Pascal

Né le 19 juin 1623 à Clermont-Ferrand, France.
Mort le 19 août 1662 à Paris.

`www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Pascal.html`

La Pascaline (1642)

Qu’est-ce que Montréal et les ordinateurs ont en commun?



`www.fi.muni.cz/usr/jkucera/pv109/sl1.htm`

Rép.: Les deux ont commencé en 1642.

Les puissances de $(x+y)$

$$
\begin{aligned}
(x+y)^0 &= 1 \\
(x+y)^1 &= x+y \\
(x+y)^2 &= x^2+2xy+y^2 \\
(x+y)^3 &= x^3+3x^2y+3xy^2+y^3 \\
(x+y)^4 &= x^4+4x^3y+6x^2y^2+4xy^3+y^4 \\
(x+y)^5 &= x^5+5x^4y+10x^3y^2+10x^2y^3+5xy^4+y^5 \\
(x+y)^6 &= x^6+6x^5y+15x^4y^2+20x^3y^3+15x^2y^4+6x \\
&\ldots = \ldots
\end{aligned}
$$

Identité de Pascal

y^k	0	1	2	3	4	5	6	7
$(x+y)^0$	1							
$(x+y)^1$	1	1						
$(x+y)^2$	1	2	1					
$(x+y)^3$	1	3	3	1				
$(x+y)^4$	1	4	6	4	1			
$(x+y)^5$	1	5	10	10	5	1		
$(x+y)^6$	1	6	15	20	15	6	1	
$(x+y)^7$	1	7	21	35	35	21	7	1

$C(n+1,k) = C(n,k-1) + C(n,k)$ pour $1 \leq k \leq n$.

Identité de Pascal

$$C(n+1,k) = C(n,k-1) + C(n,k) \text{ pour } 1 \leq k \leq n$$

DÉMONSTRATION :

$$C(n,k-1) = \frac{n!}{[n-(k-1)]!(k-1)!} = \frac{n!}{(n-k+1)(n-k)!(k-1)!}$$

$$C(n,k) = \frac{n!}{(n-k)!k!} = \frac{n!}{(n-k)k(k-1)!}$$

$$C(n,k-1) + C(n,k) = \left(\frac{1}{n-k+1} + \frac{1}{p} \right) \frac{n!}{(n-k)!(k-1)!}$$

$$= \frac{(n+1)!}{(n-k+1)!k!} = C(n+1,k)$$

Théorème du binôme

THÉORÈME: Soit x et y des variables et n un entier positif. Alors

$$\begin{aligned}(x+y)^n &= \sum_{k=0}^{n} C(n,k)x^{n-k}y^k, \\ &= \binom{n}{0}x^n y^0 + \binom{n}{1}x^{n-1}y^1 + \binom{n}{2}x^{n-2}y^2 + \cdots \\ &\quad + \binom{n}{n-1}x^1 y^{n-1} + \binom{n}{n}x^0 y^n.\end{aligned}$$

Démonstration par récurrence sur n.

Corollaire du théorème du binôme

COROLLAIRE: Soit n un entier positif. Alors

$$\sum_{k=0}^{n} C(n,k) = 2^n.$$

Preuve: Dans le théorème du binôme

$$(x+y)^n = \sum_{k=0}^{n} C(n,k) x^{n-k} y^k,$$

il suffit de prendre $x = y = 1$.

Corollaire du théorème du binôme

COROLLAIRE: Soit n un entier positif. Alors

$$\sum_{k=0}^{n}(-1)^k C(n,k) = 0.$$

Preuve: Dans le théorème du binôme

$$(x+y)^n = \sum_{k=0}^{n} C(n,k)x^{n-k}y^k,$$

il suffit de prendre $x = 1$ et $y = -1$.

Module Dénombrement Avancé

Dénombrement avancé

Définition: relation de récurrence

Une **relation de récurrence** pour la suite $\{a_n\}$ est une formule qui exprime a_n en fonction d'un ou de plusieurs termes qui le précèdent dans la suite, soit a_0, a_1, ..., a_{n-1}, pour tout entier $n \geq n_0$, où n_0 est un nombre entier non négatif.

Une suite est une **solution** d'une relation de récurrence si ces termes satisfont la relation de récurrence.

Les **conditions initiales** d'une suite spécifient les éléments qui précèdent le premier élément à partir duquel la relation de récurrence s'applique.

La relation de récurrence et les conditions initiales déterminent la suite de façon unique.

Méthodologie de résolution

Premièrement, on fait à la main les premières étapes pour bien comprendre le problème.

Deuxièmement, on définit de manière inductive l'ensemble qui nous intéresse.

Troisièmement, on dénombre, de manière inductive, le nombre d'éléments appartenant à cet ensemble.

Exemple : Nombre de déplacements pour résoudre le problème des Tours de Hanoi :

1. condition initiale : $c_1 = 1$
2. relation de récurrence : $c_n = 2c_{n-1} + 1$

Récurrence linéaire homogène de degré k

Une **relation de récurrence linéaire homogène de degré k à coefficients constants** est une relation de récurrence de la forme:

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k}$$

où $c_1, c_2, \ldots, c_k$ sont des nombres réels avec $c_k \neq 0$.

Cette relation de récurrence avec les conditions initiales

$$a_0 = C_0,\ a_1 = C_1,\ \ldots,\ a_{k-1} = C_{k-1},$$

déterminent la suite de façon unique.

Définition (suite)

Linéaire et **homogène** signifie que a_n s'écrit comme une combinaison linéaire de termes parmi a_{n-1}, a_{n-2}, etc. Contre-exemples: $a_n = 2a_{n-1} + a_{n-2}^3 + 1$ et $a_n = 3na_{n-1}$.

À coefficients constants signifie qu'ils ne dépendent pas de n.
Contre-exemple: $a_n = na_{n-1}$.

De degré k signifie que a_n est exprimé à l'aide des k éléments précédents de la suite.

Equation et racine caractéristiques

Pour la relation de récurrence linéaire homogène de degré k, $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k}$, on cherche une solution de la forme $a_n = r^n$ où r est une constante. $a_n = r^n$ est une solution si et seulement si

$$r^n = c_1 r^{n-1} + c_2 r^{n-2} + \cdots + c_k r^{n-k}.$$

Si on divise par r^{n-k} et qu'on soustrait le membre de droite du membre de gauche, on obtient l'**équation caractéristique** de la relation de récurrence:

$$r^k - c_1 r^{k-1} - c_2 r^{k-2} - \cdots - c_{k-1} r - c_k = 0.$$

Les solutions de cette équation sont appelées les **racines caractéristiques** de la relation de récurrence.

Solution: récurrence linéaire homogène de degré 2

THÉORÈME: Soit c_1 et c_2 des nombres réels. On suppose que l'équation $r^2 - c_1 r - c_2 = 0$ possède deux racines distinctes r_1 et r_2. Alors la suite $\{a_n\}$ est une solution de la relation de récurrence $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ si et seulement si $a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$ pour $n = 0, 1, 2, ...$, où α_1 et α_2 sont des constantes.

Démonstration

On doit démontrer 2 choses:

1. si r_1 et r_2 sont solutions de l'éq. car. et α_1, α_2 des constantes. alors la suite $\{a_n\}$ tq $a_n = \alpha_1 r_1^n + \alpha_2 r_2^n\}$ est solution de la relation de récurrence.
2. si la suite $\{a_n\}$ est une solution, alors $a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$ pour des constantes α_1, α_2.

Preuve:

1. Puisque r_1, r_2 sont solutions de $r^2 - c_1 r - c_2 = 0$, alors
$r_1^2 = c_1 r_1 + c_2, r_2^2 = c_1 r_2 + c_2$. On en déduit:
$c_1 a_{n-1} + c_2 a_{n-2} = c_1(\alpha_1 r_1^{n-1} + \alpha_2 r_2^{n-1}) + c_2(\alpha_1 r_1^{n-2} + \alpha_2 r_2^{n-2}) =$
$\alpha_1 r_1^{n-2}(c_1 r_1 + c_2) + \alpha_2 r_2^{n-2}(c_1 r_1 + c_2) = \ldots = a_n$

Démonstration (suite)

2. Supposons que $\{a_n\}$ est solution de la relation de récurrence, avec les conditions initiales $a_0 = C_0$ et $a_1 = C_1$. On montre qu'il existe des constantes α_1, α_2 tq $\{a_n\}$ avec $\{a_n = \alpha_1 r_1^n + \alpha_2 r_2^n\}$ satisfait ces conditions initiales.

On a donc : $a_0 = C_0 = \alpha_1 + \alpha_2$ et $a_1 = C_1 = \alpha_1 r_1 + \alpha_2 r_2$.

On en déduit:

$\alpha_1 = \frac{C_1 - C_0 r_2}{r_1 - r_2}$ et $\alpha_2 = \frac{C_0 r_1 - C_1}{r_1 - r_2}$ pour $r_1 \neq r_2$

La relation de récurrence et les conditions initiales étant satisfaites, il en ressort que $a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$

Notez que pour $r_1 = r_2$ ce théorème n'est pas vrai.

Exemple

Quelle est la solution de la relation de récurrence :

$a_n = a_{n-1} + 2a_{n-2}$ avec

$a_0 = 2$ et $a_1 = 7$?

RÉPONSE :

On applique le théorème. L'équation caractéristique $r^2 - r - 2 = 0$ a pour solutions $r = 2$ et $r = -1$.

On cherche les constantes α_1, α_2.

On a : $a_0 = 2 = \alpha_1 + \alpha_2$, $a_1 = 7 = 2\alpha_1 + (-1)\alpha_2$.

Ce qui donne: $\alpha_1 = 3$ et $\alpha_2 = -1$.

Et la solution est : $a_n = 3.2^n(-1)^n$

Solution: récurrence linéaire homogène de degré 2

THÉORÈME: Soit c_1 et c_2 des nombres réels avec $c_2 \neq 0$. On suppose que l'équation $r^2 - c_1 r - c_2 = 0$ a une racine double qui est r_0. Alors la suite $\{a_n\}$ est une solution de la relation de récurrence $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ si et seulement si $a_n = \alpha_1 r_0^n + \alpha_2 n r_0^n$ pour $n = 0, 1, 2, ...$, où α_1 et α_2 sont des constantes.

Démonstration en exercice.

Solution: récurrence linéaire homogène de degré k

THÉORÈME: Soit $c_1, c_2, ..., c_k$ des nombres réels. On suppose que l'équation caractéristique

$$r^k - c_1 r^{k-1} - c_2 r^{k-2} - \cdots - c_{k-1} r - c_k = 0$$

admet k racines distinctes $r_1, r_2, ..., r_k$. Alors la suite $\{a_n\}$ est une solution de la relation de récurrence

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k}$$

si et seulement si

$$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n + \cdots + \alpha_k r_k^n$$

pour $n = 0, 1, 2, ...$, où $\alpha_1, \alpha_2, ..., \alpha_k$ sont des constantes.

Démonstration en exercice.

Applications : Etude des complexités

Exemple : Tri par sélection. On compte le nombre de comparaisons.

1. condition initiale : $b_1 = 0$
2. a l'étape n : $b_n = n - 1 + b_{n-1}$

La résolution de la récurrence donne $b_n = n(n-1)/2$ et l'algorithme est en $\theta(n^2)$

Définition: fonction génératrice

La **fonction génératrice** de la suite de nombres réels $a_0, a_1, a_2, \ldots, a_k, \ldots$ est la série infinie

$$G(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_k x^k + \cdots = \sum_{k=0}^{\infty} a_k x^k.$$

Intérêt:
- Représenter efficacement les termes d'une suite,
- Résoudre les relations de récurrence

Définition: fonction génératrice d'une suite finie

On peut définir des **fonctions génératrices pour des suites finies** de nombres réels par extension d'une suite finie $a_0, a_1, ..., a_n$ en une suite infinie en posant $a_{n+1} = 0, a_{n+2} = 0, ...$

La fonction génératrice $G(x)$ de cette suite infinie $\{a_n\}$ est un polynôme de degré n, puisque aucun terme de la forme $a_j x^j$ avec $j > n$ n'apparaît, ce qui signifie que

$$G(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n.$$

Exemple d'une fonction génératrice

- La fonction $f(x) = 1/(1-ax)$ est la fonction génératrice de la suite $1, a, a^2, a^3, ...$, puisque

$$\frac{1}{1-ax} = 1 + ax + a^2x^2 + a^3x^3 + \cdots$$

pour $|ax| < 1$, ou de façon équivalente pour $|x| < 1/|a|$ avec $a \neq 0$.

- Cas particulier : la fonction $f(x) = 1/(1-x)$ est la fonction génératrice de la suite $1, 1, 1, 1, ...$

Addition et multiplication de fonctions génératrices

THÉORÈME: Soit $f(x) = \sum_{k=0}^{\infty} a_k x^k$ et $g(x) = \sum_{k=0}^{\infty} b_k x^k$. Alors

$$f(x) + g(x) = \sum_{k=0}^{\infty} (a_k + b_k) x^k$$

et

$$f(x)g(x) = \sum_{k=0}^{\infty} \left(\sum_{j=0}^{k} a_j b_{k-j} \right) x^k.$$

Diviser pour régner

On suppose qu'un algorithme permet de fractionner un problème de taille n en une quantité a de sous-problèmes, chacun de ces sous-problèmes étant de taille n/b.

Pour simplifier, on suppose que b est un diviseur de n. En réalité, les sous-problèmes sont souvent d'une taille égale au quotient entier le plus proche, soit inférieur, soit supérieur, soit égal à n/b.

Relation de récurrence avec fractionnement

On suppose également qu'un total de $g(n)$ opérations supplémentaires sont nécessaires quand le problème est fractionné en problèmes de taille plus réduite. Dans ce cas, si $f(n)$ représente le nombre d'opérations nécessaires pour résoudre le problème, il s'ensuit que f satisfait la relation de récurrence

$$f(n) = af(n/b) + g(n).$$

Cette équation est appelée une **relation de récurrence avec fractionnement**.

Relation de récurrence avec fractionnement

THÉORÈME: Soit f une fonction croissante qui satisfait la relation de récurrence

$$f(n) = af(n/b) + c$$

lorsque n est divisible par b, où $a \geq 1$, b est un nombre entier plus grand que 1 et où c est un nombre réel positif. Dans ce cas,

$$f(n) = \begin{cases} O(n^{\log_b a}) & \text{si } a > 1, \\ O(\log n) & \text{si } a = 1. \end{cases}$$

Une idée de la preuve est donnée page 256.

Relation de récurrence avec fractionnement

THÉORÈME: Soit f une fonction croissante qui satisfait la relation de récurrence

$$f(n) = af(n/b) + cn^d$$

lorsque $n = b^k$, où k est un entier positif, où $a \geq 1$, b est un nombre entier plus grand que 1 et c et d sont des nombres réels positifs. Alors,

$$f(n) = \begin{cases} O(n^d) & \text{si } a < b^d, \\ O(n^d \log n) & \text{si } a = b^d, \\ O(n^{\log_b a}) & \text{si } a > b^d. \end{cases}$$

Applications

Exemple: Algorithme de recherche binaire.
On compte les appels à la procédure de recherche.
Relation de récurrence: $a_1 = 2$ et $a_n = 1 + a\lfloor n/2 \rfloor$

- Si n est de la forme 2^k, la relation devient $a_{2^k} = 1 + a_{2^{(k-1)}}$ qui devient par changement de variable $b_k = 1 + b_{k-1}$ dont la solution est :

$$b_k = k + 2 \text{ et donc, } a_n = 2 + \lg n$$

- Pour un n quelconque on peut l'encadrer ainsi : $2^{k-1} < n \leq 2^k$, c-à-d $k - 1 < \lg n \leq k$. La suite $\{a_n\}$ étant croissante , on en déduit :

$$\lg n < 1 + k = a_{2^{k-1}} \leq a_n \leq a_{2^k} = 2 + k < 3 + \lg n = O(\lg n)$$

Rmq: Le théorème de la page 254 donne immédiatement le même résultat.

Module Principe d'Inclusion-Exclusion

Principe d'inclusion-exclusion

La règle de la somme

Si A et B sont des ensembles *disjoints*, alors

$$|A \cup B| = |A| + |B|.$$

Exemple: Soit A l'ensemble des lettres de l'alphabet $\{a, b, c, ..., z\}$. Soit C l'ensemble des chiffres $\{0, 1, 2, ..., 9\}$. Alors

$$|A \cup C| = |A| + |C| = 26 + 10 = 36.$$

La règle du produit

Soit A et B deux ensembles. Alors la cardinalité du produit cartésien est donnée par

$$|A \times B| = |A||B|.$$

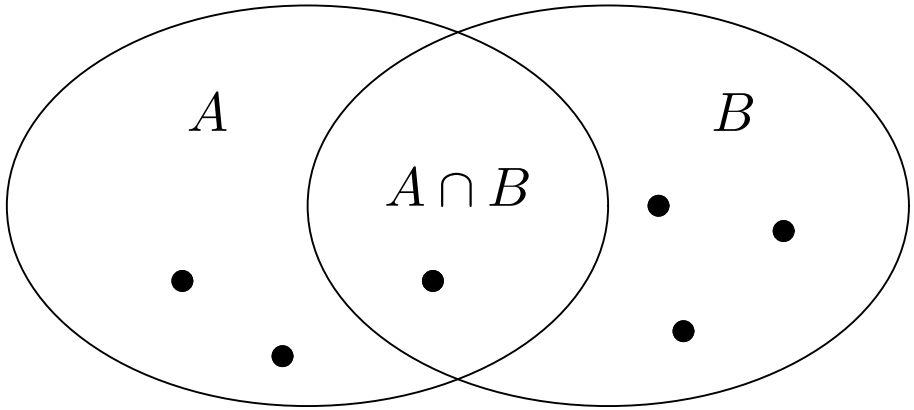
Exemple: Soit A l'ensemble des lettres de l'alphabet $\{a, b, c, ..., z\}$. Soit C l'ensemble des chiffres $\{0, 1, 2, ..., 9\}$. Alors le produit cartésien $A \times C = \{(a, c) \,|\, a \in A \wedge c \in C\}$ est de cardinalité

$$|A \times C| = |A||C| = 26 \times 10 = 260.$$

Principe d'inclusion-exclusion pour deux ensembles

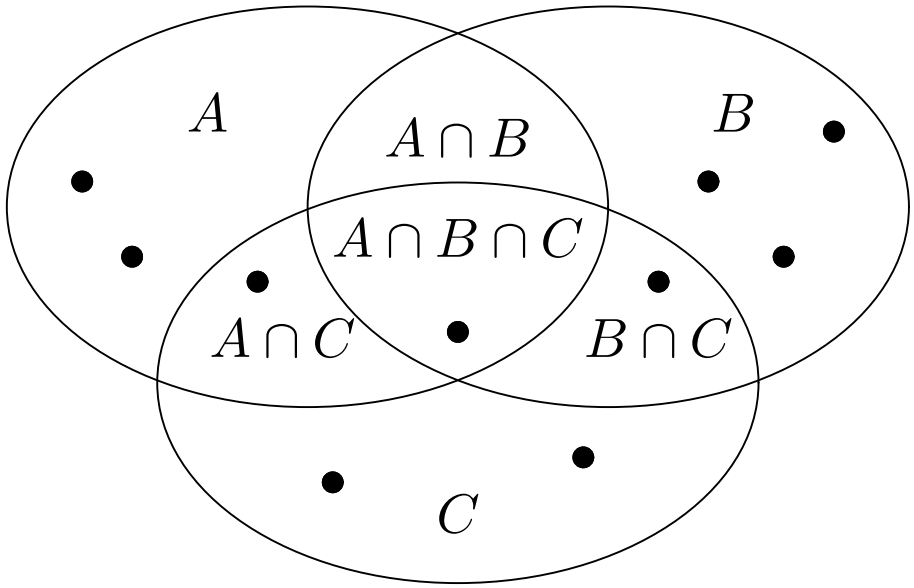
Si A et B sont deux ensembles *qui ne sont pas disjoints*, alors il ne faut pas compter les éléments qui sont dans l'intersection des deux ensembles plus d'une fois, c'est-à-dire:

$$|A \cup B| = |A| + |B| - |A \cap B|.$$



Principe d'inclusion-exclusion pour trois ensembles

$$\begin{aligned} |A \cup B \cup C| &= |A| + |B| + |C| \\ &- |A \cap B| - |A \cap C| - |B \cap C| \\ &+ |A \cap B \cap C|. \end{aligned}$$



Principe d'inclusion-exclusion pour quatre ensembles

$$
\begin{aligned}
&|A_1 \cup A_2 \cup A_3 \cup A_4| \\
&= |A_1| + |A_2| + |A_3| + |A_4| \\
&- |A_1 \cap A_2| - |A_1 \cap A_3| - |A_1 \cap A_4| - |A_2 \cap A_3| \\
&\qquad - |A_2 \cap A_4| - |A_3 \cap A_4| \\
&+ |A_1 \cap A_2 \cap A_3| + |A_1 \cap A_2 \cap A_4| + |A_1 \cap A_3 \cap A_4| \\
&\qquad + |A_2 \cap A_3 \cap A_4| \\
&- |A_1 \cap A_2 \cap A_3 \cap A_4|.
\end{aligned}
$$

Principe d'inclusion-exclusion pour n ensembles

Soit $A_1, A_2, ..., A_n$ des ensembles de cardinalité finie. Alors

$$
\begin{aligned}
|A_1 \cup A_2 \cup \cdots \cup A_n| &= \sum_{1 \le i \le n} |A_i| \\
&- \sum_{1 \le i < j \le n} |A_i \cap A_j| \\
&+ \sum_{1 \le i < j < k \le n} |A_i \cap A_j \cap A_k| \\
&- \cdots + \cdots - \cdots \\
&+ (-1)^{n+1} |A_1 \cap A_2 \cap \cdots \cap A_n|.
\end{aligned}
$$

Exercice: énoncé

Trouver combien il y a de nombres entiers positifs n'excédant pas 1000 et qui ne sont ni le carré ni le cube d'un nombre entier.

Exercice: solution

U est l'ensemble des entiers positifs n'excédant pas 1000.
$|U| = 1000$.
A est l'ensemble des carrés d'un entier n'excédant pas 1000.
$A =$
$\{1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 391, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961\}$. $|A| = 31$.

Exercice: solution (suite)

B est l'ensemble des cubes d'un entier n'excédant pas 1000.

$B = \{1, 8, 27, 64, 125, 216, 343, 512, 729, 1000\}$.

$|B| = 10$.

$A \cap B = \{1, 64, 729\}$, $|A \cap B| = 3$.

$|A \cup B| = 31 + 10 - 3 = 38$

Réponse: $|U| - |A \cup B| = 1000 - 38 = 962$.

Note historique: Ératosthène

Né en -276 à Cyrène (maintenant Shahbat en Libye).
Mort en -194 à Alexandrie en Égypte.
Il fut le troisième bibliothécaire de la bibliothèque d'Alexandrie. Il calcula, avec une précision raisonnable, le rayon de la terre.

`www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Eratosthenes.html`

Le crible d'Ératosthène

Le crible d'Ératosthène sert à trouver les nombres premiers qui n'excèdent pas un nombre entier positif donné.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Le crible d'Ératosthène — 2

Les entiers divisible par 2, autres que 2, sont soulignés.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Le crible d'Ératosthène — 3

Les entiers divisible par 3, autres que 3, sont soulignés.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Le crible d'Ératosthène — 4

Les entiers divisible par 4, autres que 4, sont soulignés.

Cette étape n'est pas nécessaire. 4 étant déjà souligné, ce n'est donc pas un nombre premier.

4 étant déjà souligné, c'est un multiple de nombres premiers. Alors tous les multiples de 4 sont aussi des multiples de nombres premiers. Et donc tous les multiples de 4 sont déjà soulignés.

Le crible d'Ératosthène — 5

Les entiers divisible par 5, autres que 5, sont soulignés.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Le crible d'Ératosthène — 6

Les entiers divisible par 6, autres que 6, sont soulignés.

Cette étape n'est pas nécessaire. 6 étant déjà souligné, ce n'est donc pas un nombre premier et tous les multiples de 6 sont déjà soulignés.

Le crible d'Ératosthène — 7

Les entiers divisible par 7, autres que 7, sont soulignés.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Le crible d'Ératosthène — 8, 9, 10, 11

Les entiers divisible par 8, 9 et 10, autres que 8, 9 et 10, sont déjà soulignés car ces entiers ne sont pas des nombres premiers.

Les entiers divisible par 11, autres que 11, sont soulignés. Cette étape n'est pas nécessaire car $11 > 10 = \sqrt{100}$.

Le crible d'Ératosthène

Les entiers en **rouge** sont les nombres premiers ≤ 100.

1	**2**	**3**	4	**5**	6	**7**	8	9	10
11	12	**13**	14	15	16	**17**	18	**19**	20
21	22	**23**	24	25	26	27	28	**29**	30
31	32	33	34	35	36	**37**	38	39	40
41	42	**43**	44	45	46	**47**	48	49	50
51	52	**53**	54	55	56	57	58	**59**	60
61	62	63	64	65	66	**67**	68	69	70
71	72	**73**	74	75	76	77	78	**79**	80
81	82	**83**	84	85	86	87	88	**89**	90
91	92	93	94	95	96	**97**	98	99	100

Le nombre de premiers ≤ 100

Aurait-il été possible de connaître le nombre de premiers ≤ 100 sans les énumérer avec le crible d'Ératosthène?

Le nombre de premiers ≤ 100 est égal à 100 moins le nombre de nombres composés moins un car un n'est pas considéré comme un nombre premier.

Soit U l'ensemble des nombres entiers ≤ 100.
Soit P l'ensemble des nombres premiers ≤ 100.
Soit P_2 l'ensemble des multiples de 2 ≤ 100.
Soit P_3 l'ensemble des multiples de 3 ≤ 100.
Soit P_5 l'ensemble des multiples de 5 ≤ 100.
Soit P_7 l'ensemble des multiples de 7 ≤ 100.

$$|P| = |U| - |P_2 \cup P_3 \cup P_5 \cup P_7| - 1 + 4.$$

Le nombre d'entiers composés ≤ 100

En appliquant le principe d'inclusion-exclusion, le nombre d'entiers ≤ 100 qui sont divisibles par 2, 3, 5 ou 7 est donné par:

$$
\begin{aligned}
&|P_2 \cup P_3 \cup P_5 \cup P_7| \\
&= |P_2| + |P_3| + |P_5| + |P_7| \\
&- |P_2 \cap P_3| - |P_2 \cap P_5| - |P_2 \cap P_7| - |P_3 \cap P_5| \\
&\quad - |P_3 \cap P_7| - |P_5 \cap P_7| \\
&+ |P_2 \cap P_3 \cap P_5| + |P_2 \cap P_3 \cap P_7| + |P_2 \cap P_5 \cap P_7| \\
&\quad + |P_3 \cap P_5 \cap P_7| \\
&- |P_2 \cap P_3 \cap P_5 \cap P_7|.
\end{aligned}
$$

Le nombre d'entiers composés ≤ 100

Le nombre d'entiers ≤ 100 qui sont divisibles par un nombre entier n est donné par $\lfloor 100/n \rfloor$.

$$
\begin{aligned}
&|P_2 \cup P_3 \cup P_5 \cup P_7| \\
&= \left\lfloor \frac{100}{2} \right\rfloor + \left\lfloor \frac{100}{3} \right\rfloor + \left\lfloor \frac{100}{5} \right\rfloor + \left\lfloor \frac{100}{7} \right\rfloor \\
&- \left\lfloor \frac{100}{2 \cdot 3} \right\rfloor - \left\lfloor \frac{100}{2 \cdot 5} \right\rfloor - \left\lfloor \frac{100}{2 \cdot 7} \right\rfloor - \left\lfloor \frac{100}{3 \cdot 5} \right\rfloor - \left\lfloor \frac{100}{3 \cdot 7} \right\rfloor - \\
&+ \left\lfloor \frac{100}{2 \cdot 3 \cdot 5} \right\rfloor + \left\lfloor \frac{100}{2 \cdot 3 \cdot 7} \right\rfloor + \left\lfloor \frac{100}{2 \cdot 5 \cdot 7} \right\rfloor + \left\lfloor \frac{100}{3 \cdot 5 \cdot 7} \right\rfloor \\
&- \left\lfloor \frac{100}{2 \cdot 3 \cdot 5 \cdot 7} \right\rfloor .
\end{aligned}
$$

Le nombre d'entiers composés ≤ 100

Le nombre d'entiers ≤ 100 qui sont divisibles par 2, 3, 5 ou 7 est donné par:

$$\begin{aligned} |P_2 \cup P_3 \cup P_5 \cup P_7| &= 50 + 33 + 20 + 14 \\ &\quad - 16 - 10 - 7 - 6 - 4 - 2 \\ &\quad + 3 + 2 + 1 + 0 \\ &\quad - 0 \\ &= 78. \end{aligned}$$

Le nombre de premiers ≤ 100

Soit U l'ensemble des nombres entiers ≤ 100.
Soit P l'ensemble des nombres premiers ≤ 100.
Soit P_2 l'ensemble des multiples de 2 ≤ 100.
Soit P_3 l'ensemble des multiples de 3 ≤ 100.
Soit P_5 l'ensemble des multiples de 5 ≤ 100.
Soit P_7 l'ensemble des multiples de 7 ≤ 100.

$$
\begin{aligned}
|P| &= |U| - |P_2 \cup P_3 \cup P_5 \cup P_7| - 1 + 4 \\
|P| &= 100 - 78 - 1 + 4 \\
|P| &= 25
\end{aligned}
$$

Autre forme du principe d'inclusion-exclusion

On cherche le nombre d'éléments qui, à l'intérieur d'un ensemble, n'ont aucune des n propriétés $P_1, P_2, ..., P_n$.

Soit N le nombre d'éléments de l'ensemble considéré.
Soit A_i, le sous-ensemble contenant les éléments qui ont la propriété P_i.
Le nombre d'éléments qui ont toutes les propriétés $P_{i_1}, P_{i_2}, ..., P_{i_k}$ est représenté par $N(P_{i_1} P_{i_2} \dots P_{i_k})$.
Sous forme d'ensembles, cela s'écrit

$$|A_{i_1} \cap A_{i_2} \cap \cdots \cap A_{i_k}| = N(P_{i_1} P_{i_2} \dots P_{i_k}).$$

Autre forme du principe d'inclusion-exclusion (suite)

Le nombre d'éléments ne contenant aucune des propriétés $P_1, P_2, ..., P_n$ est représenté par $N(P_1' P_2' \cdots P_n')$.
Ainsi, le nombre d'éléments qui, à l'intérieur d'un ensemble, n'ont aucune des n propriétés $P_1, P_2, ..., P_n$ est donné par

$$N(P_1' P_2' \cdots P_n') = N - |A_1 \cup A_2 \cup \cdots \cup A_n|$$

Autre forme du principe d'inclusion-exclusion (suite)

Et par le principe d'inclusion-exclusion,

$$
\begin{aligned}
N(P_1' P_2' \cdots P_n') &= N \;-\; |A_1 \cup A_2 \cup \cdots \cup A_n| \\
&= N \;-\; \sum_{1 \le i \le n} N(P_i) \\
&\quad + \sum_{1 \le i < j \le n} N(P_i P_j) \\
&\quad - \sum_{1 \le i < j < k \le n} N(P_i P_j P_k) \\
&\quad + \cdots - \cdots + \cdots \\
&\quad + (-1)^n N(P_1 P_2 \cdots P_n).
\end{aligned}
$$

Dénombrement des fonctions surjectives

Combien y a-t-il de fonctions surjectives d'un ensemble à 6 éléments dans un ensemble de 3 éléments?

Le nombre total fonctions surjectives d'un ensemble à 6 éléments dans un ensemble de 3 éléments est le nombre total de fonctions moins celles qui ne sont pas surjectives.

On a deux nouveaux problèmes:
1) Quel est le nombre total de fonctions d'un ensemble à 6 éléments dans un ensemble de 3 éléments.
2) Quel est le nombre total de fonctions de cet ensemble qui ne sont pas surjectives.

Dénombrement des fonctions

Combien de fonctions différentes y a-t-il d'un ensemble à m éléments dans un ensemble à n éléments.

À chacun des m éléments dans le domaine, on associe l'un des n éléments du codomaine, au choix. Selon le principe du produit, il y a

$$\underbrace{n \cdot n \cdot \cdots \cdot n}_{m \text{ fois}} = n^m$$

fonctions différentes.

Dénombrement des fonctions non surjectives

Soit P_i la propriété voulant que b_i ne soit pas dans l'image.

Le nombre de fonctions qui ont la propriété P_1 est le nombre de fonctions d'un ensemble à 6 éléments dans un ensemble de 2 éléments.

$$N(P_1) = N(P_2) = N(P_3) = 2^6 = 64.$$

Le nombre de fonctions qui ont deux propriétés P_1 et P_2 est le nombre de fonctions d'un ensemble à 6 éléments dans un ensemble de 1 élément.

$$N(P_1P_2) = N(P_1P_3) = N(P_2P_3) = 1^6 = 1.$$

Dénombrement des fonctions non surjectives

Le nombre de fonctions qui ont les trois propriétés P_1, P_2 et P_3 est le nombre de fonctions d'un ensemble à 6 éléments dans un ensemble de 0 élément.

$$N(P_1P_2P_3) = 0^6 = 0.$$

Le nombre de fonctions surjectives est donc

$$\begin{aligned} N(P_1'P_2'P_3') &= N - (N(P_1) + N(P_2) + N(P_3)) \\ &+ (N(P_1P_2) + N(P_1P_3) + N(P_2P_3)) - N(\\ &= 3^6 - C(3,1)2^6 + C(3,2)1^6 - C(3,3)0^6 \\ &= 729 - 192 + 3 = 540. \end{aligned}$$

Dénombrement des fonctions surjectives

THÉORÈME: Soit m et n des nombres entiers positifs avec $m \geq n$. Alors, il y a

$$n^m - C(n,1)(n-1)^m + C(n,2)(n-2)^m - \cdots + (-1)^{n-1} C(n,n-1) \cdot 1^m$$

fonctions surjectives d'un ensemble à m éléments dans un ensemble à n éléments.

Définition: dérangement

Un **dérangement** est une permutation d'objets qui ne laisse aucun objet dans sa position originale.

Exemple: La permutation 21453 est un dérangement de 12345. Par contre 21543 n'en est pas un.

Nombre de dérangements

On note D_n le **nombre de dérangements** de n objets. Par exemple, $D_3 = 2$, puisque les dérangements de 123 sont 231 et 312.

Le nombre de dérangement d'un ensemble contenant n éléments est

$$D_n = n! \left(1 - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \cdots + (-1)^n \frac{1}{n!}\right).$$

Idée de la preuve : On appelle P_i la propriété: "*Le dérangement fixe le nombre* i". On calcule $D_n = N(P_1' P_2' ... P_n')$ où P_i' signifie "n'a pas la propriété P_i".