SINF1250: Mathématiques pour l'Informatique

Année 2009-2010

Laurence A. Wolsey

February 1, 2010

Contents

1 Ensembles, Relations Binaires et Fonctions **7**
- 1.1 Ensembles . . . 7
- 1.2 Relation Binaire . . . 9
- 1.3 Composition de relations binaires . . . 10
- 1.4 Inverse d'une relation binaire . . . 11
- 1.5 Fonctions de $A \rightarrow B$ . . . 11
- 1.6 Propriétés importantes des fonctions . . . 12
- 1.7 Relations Binaires sur A . . . 14
- 1.8 Relation d'équivalence . . . 14
- 1.9 Ordre Partiel (Partially Ordered Set) . . . 16
- 1.10 Relation 1-aire . . . 18
- 1.11 Ensembles finis et infinis . . . 18

2 Logique des Propositions: une Introduction **21**
- 2.1 Les Buts . . . 21
- 2.2 Introduction . . . 23
- 2.3 Le langage (Syntaxe) . . . 24
- 2.4 Le sens d'une phrase (Sémantique) . . . 25
- 2.5 Propriétés des phrases . . . 27
- 2.6 Vérification par Tables de Vérité . . . 28
- 2.7 Utilisation formelle des equivalences et implications . . . 30
- 2.8 Le nombre de propositions distincts . . . 32
- 2.9 Une algèbre booléenne: les propriétés . . . 33

3 Techniques de Démonstration **35**
- 3.1 Inférence Logique . . . 35
- 3.2 Approches Directe, Indirecte, par l'Absurde . . . 35
  - 3.2.1 Une démonstration directe . . . 35
  - 3.2.2 Une démonstration indirecte . . . 36
  - 3.2.3 Preuve par l'absurde . . . 36
- 3.3 Preuves Constructives et Non Constructives . . . 37
  - 3.3.1 Preuve d'existence constructive $(\forall\ n)P(n)$ . . . 37
  - 3.3.2 Preuve d'existence non constructive . . . 37
- 3.4 Les Entiers et le Principe du Bon Ordre . . . 38

3.5 Principe d'Induction Mathématique . . . 38
3.5.1 Principe d'induction . . . 38
3.5.2 Principe des Tiroirs (Pigeonhole Principle) . . . 40
3.5.3 Principe de diagonalisation . . . 41
3.5.4 Preuve . . . 41

4 Dénombrement **43**
4.1 Unions Disjointes et Produits Cartésiens . . . 43
4.2 Permutations et Combinaisons . . . 44
4.3 Théorème du Binome . . . 46
4.4 Permutations et Combinaisons avec Remise . . . 47
4.5 Principe Inclusion-Exclusion . . . 48
4.5.1 Dénombrement des Fonctions Surjectives . . . 50
4.5.2 Dérangements . . . 50

5 Relations de Récurrence **53**
5.1 Etablir une Relation de Récurrence . . . 53
5.2 Solutions des Relations de Récurrence . . . 56
5.3 Relations de Récurrence avec Fractionnement . . . 59
5.4 Fonctions Génératrices . . . 60

6 Algorithmes et Complexité **65**
6.1 Algorithme de Fouille Binaire . . . 65
6.2 Le plus grand diviseur de deux entiers . . . 66
6.2.1 Algorithme d'Euclide 1 . . . 66
6.2.2 Une Preuve Rapide . . . 67
6.2.3 Algorithme 2 . . . 68
6.2.4 Efficacité de l'Algorithme . . . 69
6.3 Remarque sur la complexité . . . 71
6.4 Programmes Récursives et Itératives . . . 71
6.4.1 Algorithme d'Euclide . . . 71
6.4.2 Calcul du factorielle . . . 71
6.4.3 Nombres de Fibonnaci . . . 72
6.5 Introduction à la Programmation Dynamique . . . 74
6.5.1 Multiplication de Matrices . . . 78
6.5.2 Problème . . . 78
6.5.3 Observation . . . 79

7 Connexité et Arbres **81**
7.1 Notions de base . . . 81
7.2 Connexité . . . 84
7.3 Coupe (edge cut set) et Cocycle . . . 86
7.4 Arbres . . . 87
7.5 Arbre de poids minimum . . . 90
7.5.1 Algorithme 1 . . . 90
7.5.2 Présentation formelle . . . 91

7.5.3 Temps d'exécution 91
7.5.4 Algorithme de PRIM 92

8 Chemins **95**
8.1 Graphes et Isomorphisme 95
8.2 La matrice d'adjacence de G. 96
8.3 Le degré des nœuds 98
8.4 Chemins, parcours et cycles 98
8.5 Le Problème du Plus Court Chemin d'un Graphe 100
8.6 Digraphes 100
8.7 Chemin de Coût Minimum d'un Digraphe 101

9 Chemins et Tours **105**
9.1 Modélisation 105
9.2 Ordre Topologique et Digraphes Acycliques 107
9.3 Le Problème du Plus Court Chemin 109
9.4 PCC dans un Digraphe Acyclique 110
9.5 PCC avec Distances Non-negatives 111
9.6 Tours Eulériens et Hamiltoniens 112
9.7 Théorie de la complexité *(une brève introduction)* 115

10 Encore des Graphes **117**
10.1 Graphes Planaires 117
10.2 Formule d'Euler 119
10.3 Coloriage des Graphes 120
10.4 Encore des Arbres 121
10.4.1 Propriété des arbres 122
10.5 Arbre de Feuille Binaire (Retour) 122
10.6 Parcours d'un arbre 123
10.6.1 Systèmes d'Adressage Universel 124
10.6.2 Représentations des Expressions 125

11 Ordres Partiels et Treillis **129**
11.1 Ordres Partiels 129
11.2 Ordre de classement 130
11.3 Diagramme de Hasse 131
11.3.1 Elément maximal, maximum, ... 131
11.4 Treillis 134
11.5 Tri topologique de nouveau 138

Chapter 1

Ensembles, Relations Binaires et Fonctions

1.1 Ensembles

Definition 1.1 Les objets qui constituent un *ensemble* sont ses *éléments*.
Deux ensembles sont *égaux* ssi ils contiennent les mêmes éléments.
A est un *sous-ensemble* de B ssi $(\forall\ x)$ si $x \in A$, alors $x \in B$. $A \subseteq B$.
Il existe un *ensemble vide* $\{\ \} = \emptyset$ qui ne contient aucun élément.
L' ensemble de sous-ensembles de S est un ensemble noté $\mathcal{P}(\mathcal{S})$.

Opérations sur les ensembles.

Definition 1.2 Soit A et B deux ensembles. *L'union*, notée $A \cup B$, est l'ensemble qui contient les éléments soit de A, soit de B.
Intersection. $A \cap B = \{x : x \in A \text{ et } x \in B\}$.
Disjoint. $A \cap B = \emptyset$.
Différence. $A \setminus B = \{x : x \in A \text{ et } x \notin B\}$.
L'ensemble universel U. On considère que tous les ensembles sont des sous-ensembles de U. Alors le complément de A, noté $\bar{A}$, est défini par $\bar{A} = U \setminus A$.

Propriétés de base

$A \cup \phi = A$	*Identité*
$A \cap U = A$	
$A \cup U = U$	*Domination*
$A \cap \phi = \phi$	
$A \cup A = A$	*Idempotence*
$A \cap A = A$	
$\neg(\neg A) = A$	*Complémentarité*
$A \cup B = B \cup A$	*Commutativité*
$A \cap B = B \cap A$	
$A \cup (B \cup C) = (A \cup B) \cup C$	*Associativité*
$A \cap (B \cap C) = (A \cap B) \cap C$	
$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$	*Distributivité*
$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	
$\neg(A \cup B) = \neg A \cap \neg B$	*Lois de De Morgan*
$\neg(A \cap B) = \neg A \cup \neg B$	
$A \subseteq A \cup B$	
$A \cap B \subseteq A$	
$A \cup \neg A = U$	

Considérons par exemple la distributivité.

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

Démonstration 1. On va utiliser $X = Y$ ssi $X \subseteq Y$ et $Y \subseteq X$. En plus, pour montrer que $A \cup B \subseteq C$, on va montrer que $A \subseteq C$ et $B \subseteq C$.

i) $X \subseteq Y$. Soit $x \in A \cup (B \cap C)$.
a). $A \subseteq A \cup B$ et $A \subseteq A \cup C$. Donc $A \subseteq (A \cup B) \cap (A \cup C)$.
b). $(B \cap C) \subseteq B \subseteq (A \cup B)$ et $(B \cap C) \subseteq C \subseteq (A \cup C)$. Donc $(B \cap C) \subseteq (A \cup B) \cap (A \cup C)$.
Alors combinant a) et b), $A \cup (B \cap C) \subseteq (A \cup B) \cap (A \cup C)$
ii). $Y \subseteq X$. Pour montrer que $Y \subseteq X$, on verifie que si $x \in Y$, alors $x \in X$.
$x \in (A \cup B) \cap (A \cup C)$ impliquent que $x \in (A \cup B)$ et $x \in (A \cup C)$. Donc soit $x \in A$, soit $x \notin A$ et alors $x \in B$ et $x \in C$. Donc $x \in A \cup (B \cap C)$.

Démonstration 2. Tables d'appartenance.
Autrement dit, on considère cas par cas.

A	B	C	$B \cap C$	$A \cup (B \cap C)$	$A \cup B$	$A \cup C$	$(A \cup B) \cap (A \cup C)$
1	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	0	0	0	1	1	1	1
0	1	1	1	1	1	1	1
0	1	0	0	0	1	0	0
0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	0

Par exemple la ligne 6 indique que "Si $x \notin A$, $x \in B$ et $x \notin C$, alors $x \notin B \cap C, x \notin A \cup (B \cap C)$, $x \in A \cup B$, etc. Donc pour le cas où $x \notin A$, $x \in B$ et $x \notin C$, x n'est ni dans $A \cup (B \cap C)$ ni dans $(A \cup B) \cap (A \cup C)$.

On voit que les colonnes $A \cup (B \cap C)$ et $(A \cup B) \cap (A \cup C)$ sont identiques pour tous les cas, et donc on a un cas par cas verification de l'égalité.

1.2 Relation Binaire

Soit $A = \{a_1 \ \ldots \ a_n\}$ ou $\{a_1 \ \ldots \ a_n \ \ldots\}$
et $B = \{b_1 \ \ldots \ b_n\}$ ou $\{b_1 \ \ldots \ b_n \ \ldots\}$

Definition 1.3 Le *produit cartésien* $A \times B$ est l'ensemble des paires ordonnées

$$(a_i, b_j) \text{ où } a_i \in A \ \text{ et } \ b_j \in B$$

Definition 1.4 Une *relation binaire* α sur $A \times B$ est un sous-ensemble $\mathcal{R}_\alpha$ de $A \times B$.

On note $a_i \ \alpha \ b_j \iff (a_i, b_j) \in \mathcal{R}_\alpha$.

> (*En d'autres termes, tout ensemble de paires ordonnées inclus dans $A \times B$ définit une relation binaire de $A \to B$)*

Example 1.1

$$\begin{aligned} A &= \{1, 2, 3\} \\ B &= \{1', 2', 3', 4'\} \\ \mathcal{R}_\alpha &= \Big\{(1, 3'), (2, 4'), (1, 1'), (3, 2')\Big\} \end{aligned}$$

Example 1.2

$$\begin{aligned} A &= Z \\ B &= Z^+ \\ \mathcal{R}_\alpha &= \Big\{(0,0), (-1,1), (1,1), (-2,4), (2,4), \ldots\Big\} = \Big\{(x, x^2) \ : \ x \in Z\Big\} \end{aligned}$$

Example 1.3

$a\alpha b$ si et seulement si a est le père de b

On peut représenter α par une matrice M à coefficients $0-1$ où l'élément a_i de A est représenté par la ligne i et l'élément b_j de B est représenté par la colonne j. On définit les éléments de la matrice M de la façon suivante:

$$\begin{aligned} m_{ij} &= 1 \text{ si } a_i \ \alpha \ b_j \\ &= 0 \text{ sinon.} \end{aligned}$$

Pour l'exemple 1.1, la matrice M est:

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

On peut aussi représenter α par un digraphe (graphe orienté) bipartite (c-à-d il y a une partition $A \cup B$ des noeuds et chaque arc va de A à B).

$$G = (\overbrace{A \cup B}^{\text{Sommets}}, \overbrace{\mathrm{R}_\alpha}^{\text{arcs}})$$

Pour l'exemple 1.1, le digraphe est le suivant :

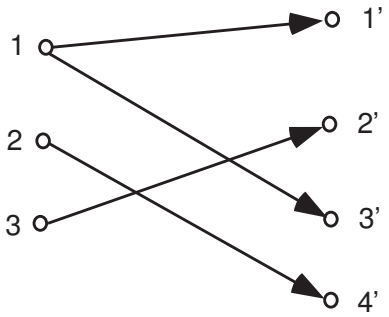


Figure 1.1:

1.3 Composition de relations binaires

Soit $\alpha : A \to B$ et $\beta : B \to C$ deux relations binaires, alors

Definition 1.5 $\alpha\beta$ est une relation binaire, appelée *composée de* α et β, telle que

$$a(\alpha\beta)c \Longleftrightarrow \exists b \in B \text{ t.q. } a\alpha b \text{ et } b\beta c$$

Proposition 1.1 *Etant données des relations binaires α, β, γ telles que $(\alpha\beta)\gamma$ est définie, alors*

$$(\alpha\beta)\gamma \;=\; \alpha(\beta\gamma)$$

Cette propriété est connue sous le nom d'*associativité de la composée*.
Preuve $\alpha : A \to B \;\;;\;\; \beta : B \to C \;\;;\;\; \gamma : C \to D$

$$\begin{aligned} a(\alpha\beta)\gamma d &\iff \exists c \text{ t.q. } a(\alpha\beta)c \text{ et } c\gamma d \\ &\iff \exists c \text{ t.q. } c\gamma d \text{ et } \exists b \text{ t.q. } a\alpha b \text{ et } b\beta c \\ &\iff \exists b, c \text{ t.q. } a\alpha b,\; b\beta c,\; c\gamma d \\ &\iff \exists b \text{ t.q. } a\alpha b,\; b(\beta\gamma)d \\ &\iff a\alpha(\beta\gamma)d \end{aligned}$$

Il est possible de démontrer cette propriété en joignant les digraphes bipartites représentant α et β (Voir Figure 1.2). Sur ce digraphe, on voit que $a(\alpha\beta)c \iff$ il existe un chemin orienté dans le graphe de a vers c. La propriété d'associativité suit immédiatement de cette constatation.

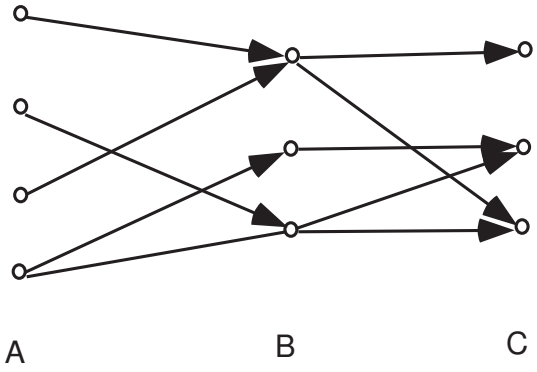


Figure 1.2:

1.4 Inverse d'une relation binaire

Definition 1.6 α^{-1}, l'*inverse* de α, est une relation binaire de $B \to A$ définie par $b\alpha^{-1}a \iff a\alpha b$

(**c.à.d.** qu'il suffit pour obtenir α^{-1} d'inverser la direction des arcs du digraphe G représentant α ou de transposer la matrice M représentant α)

1.5 Fonctions de $A \to B$

On introduit la notion de fonction car les relations binaires ont une structure trop pauvre pour pouvoir être exploitée de façon intéressante.

Definition 1.7 Une relation binaire $\alpha : A \to B$ est une *fonction de A vers B* si $\forall a \in A \; \exists! \; b \in B$ tel que $a\alpha b$[1].

Généralement on note une fonction $f : A \to B$ par $a \to f(a)$. La notation $f(a)$ n'est pas ambiguë et $f(a)$ est bien un seul élément de B. Dans ce cas,

- A est le *domaine* de f
- B est le *codomaine* de f
- $f(A) = \{f(a) : a \in A\}$ est *l'image* de A par f (*range* of A en anglais)

Nous nous intéressons aussi aux *fonctions partielles* $f : A \to B$ où f n'est définie que sur une partie $A' \subset A$.

Autrement dit $f : A \to B$ représente la relation binaire $(a, f(a))$ pour tout $a \in A$. $f' = f|A' : A' \to B$ est appelée la *restriction* de f à A', si elle représente la relation binaire $(a, f(a))$ pour tout $a \in A'$.

En termes du digraphe représentant une relation binaire, α est une fonction **ssi** il existe exactement un arc quittant chaque noeud $a_i \in A$.

On parlera de *fonction sur un ensemble* pour $f : A \to A$.

1.6 Propriétés importantes des fonctions

(1) $f : A \to B$ est *surjective* (*onto*) si tout $b \in B$ est l'image d'au moins un $a \in A$ (au moins un arc arrive à chaque noeud $b_j \in B$) $(f(A) = B)$

(2) $f : A \to B$ est *injective* (*one-to-one*) si $a_1 \neq a_2 \Rightarrow f(a_1) \neq f(a_2)$ (au plus un arc arrive à chaque noeud $b_j \in B$)

(3) $f : A \to B$ est *bijective* si f est à la fois injective et surjective

(4) Si A est fini, une bijection $f : A \to A$ est une *permutation*.

Proposition 1.2 *Soit f une fonction de A vers B. Alors f est une bijection si et seulement si la relation binaire f^{-1} est une fonction de B vers A.*

Preuve On utilisera le digraphe représentant la relation binaire f.

- Si f est une bijection, alors il doit y avoir *exactement* un arc qui arrive à chaque noeud b de B. Si on inverse l'orientation des arcs pour obtenir f^{-1}, il y aura exactement un arc quittant chaque noeud $b \in B$. Donc f^{-1} est une fonction.
- Si f^{-1} est une fonction, alors chaque sommet de B est incident à exactement un arc, c.à.d. que la fonction f est à la fois injective et surjective.

[1] ! veut dire unique; donc $\exists!$ = *il existe un et un seul.*

Example 1.4

$f: Z \to Z_+: x \to x^2$ est une fonction qui est ni injective, ni surjective.

Example 1.5

$f: Z_+ \to Z_+: x \to x^2$ est une fonction qui est injective, mais pas surjective.

Example 1.6

$f: Z_+ \to Z_+: x \to \lfloor \frac{x}{2} \rfloor$ est une fonction qui est surjective, mais pas injective.[2]

Example 1.7

$A = B =$ ensemble des hommes vivants : $f(x)$ est la relation père de x.

Sous cette forme, la relation f de A vers A n'est pas une fonction car tout homme vivant ne possède plus nécessairement son père vivant.

Soit $B' = A \cup \{d\}$ et f' une relation de A à B' définie par

$f'(x) =$ d si le père de x est mort

$f'(x) =$ $f(x)$ sinon.

Dans ce cas, f' est une fonction qui n'est ni injective (un père peut avoir plusieurs fils) ni surjective (un homme vivant peut ne pas avoir d'enfants).

Example 1.8 *Soient $f: A \to B$ une fonction partielle définie sur $A' \subset A$ et $B' = B \cup \{w\}$ (où w = indéfini).*

Si on définit

$$\begin{aligned} f'(a) &= w \text{ si } a \in A \backslash A' \\ f'(a) &= f(a) \text{ si } a \in A' \end{aligned}$$

alors f' est une fonction : $A \to B'$.

Example 1.9

$$\begin{aligned} A = B &= \{0, 1, \ldots, q-1\} \\ f(x) &= rx \pmod q \text{ où } r \in A \backslash \{0\} \end{aligned}$$

par exemple $(r = 3,\ q = 4)$ $f(x) = 3x \pmod 4$

Sous quelles conditions une telle fonction est-elle inversible (bijection)?

x	f
0	0
1	3
2	2
3	1

[2] $\lfloor x \rfloor$ =partie entière de x.

Example 1.10 *Soit $f : A \to B$ et $g : B \to C$. Alors $g \circ f : A \to C$ est la composée des fonctions f et g.*

Montrer que

- $g \circ f$ est surjective si f et g sont surjectives;
- $g \circ f$ est injective si f et g sont injectives.

1.7 Relations Binaires sur A

$$\alpha : A \to A \iff \mathcal{R}_\alpha \subseteq A \times A$$

(1)	$\alpha : A \to A$	est *réflexive*	ssi $a\alpha a \ \ \forall a \in A$
(2)	$\alpha : A \to A$	est *symétrique*	ssi $a\alpha b \ \Rightarrow b\alpha a \ \ \forall a, b \in A$
(3)	$\alpha : A \to A$	est *antisymétrique*	ssi $a\alpha b$ et $b\alpha a \Rightarrow a = b \ \forall a, b \in A$
(4)	$\alpha : A \to A$	est *transitive*	ssi $a\alpha b$ et $b\alpha c \Rightarrow a\alpha c \ \ \forall a, b, c \in A.$

Example 1.11 *$A = Z_+ \ \ x\alpha y \Leftrightarrow x \geq y$: relation réflexive, antisymétrique, transitive.*

Example 1.12 *$A = Z_+ \ \ x\alpha y \Leftrightarrow x > y$: relation non réflexive, non symétrique, transitive, antisymétrique.*

Example 1.13 *$A = Z \ \ x\alpha y \Leftrightarrow |x - y| \leq 1$: relation symétrique, réflexive, non-transitive.*

1.8 Relation d'équivalence

Definition 1.8 *Une relation d'équivalence* est une relation sur un ensemble A qui est réflexive, symétrique, et transitive.

Example 1.14 *$A = Z$.*

$$\begin{aligned}
x\mathcal{R}y \Leftrightarrow \quad & x - y = 11k \text{ (pour un entier } k \text{ non fixé)} \\
\textit{Réflexive} \quad & x - x = 11 \cdot 0 \\
\textit{Symétrique} \quad & x - y = 11k \Rightarrow y - x = 11(-k) \\
\textit{Transitive} \quad & x - y = 11k_1;\ y - z = 11k_2 \Rightarrow x - z = 11(k_1 + k_2)
\end{aligned}$$

$$E(x) = \{y \in A \ : \ x\mathcal{R}y\} \ \triangleq \text{ la classe d'équivalence de } x$$

Example 1.15 *La classe d'équivalence de 14 dans l'exemple ci-dessus est* $\{3, 14, 25, \ldots, -8, -19, \ldots\}$

On observe d'abord que si $a_1\mathcal{R}a_2$, alors $a_1, a_2 \in E(a_1)$.

Proposition 1.3 *Soit α une relation d'équivalence sur A. Tout membre $a \in A$ appartient à une et une seule classe d'équivalence.*

Preuve Si $E(a_1) \cap E(a_2) \neq \phi$, il existe a tel que $a \in E(a_1)$ et $a \in E(a_2)$. Donc par définition, $a_1 \mathcal{R} a$ et $a_2 \mathcal{R} a$.

Par symétrie $a \mathcal{R} a_2$ et par transitivité

$$a_2 \in E(a_1) \Rightarrow E(a_2) \subseteq E(a_1).$$

En sens inverse on a

$$E(a_1) \subseteq E(a_2) \Rightarrow E(a_1) = E(a_2).$$

Donc un élément a ne peut pas appartenir à 2 classes différentes.

De plus $a \in E(a)$, ce qui implique qu'un élément appartient à exactement une classe. ∎

La conséquence de cette proposition est que l'ensemble des classes d'équivalence définit une *partition* de A. (Sous-ensembles non vides de A, disjoints 2 à 2 et dont l'union est A) L'inverse est également vrai: une partition d'un ensemble A détermine une relation d'équivalence sur A.

Example 1.16 *Soit*

$$S = \{(a, b) \ : \ a, b \in Z; \ b \neq 0\}.$$

Soit $\mathcal{R}$ la relation de S à S telle que $(a, b)\mathcal{R}(c, d) \Longleftrightarrow ad = bc$.

Alors la classe d'équivalence $E(a, b)$ représente toutes les fractions $\frac{c}{d}$ telles que $\frac{c}{d} = \frac{a}{b}$.

$$E((4, 6)) = \{(2, 3), (-2, -3), (4, 6), (-4, -6), (6, 9), (-6, -9), \cdots\}.$$

$$\begin{aligned} \textit{Alors } \mathcal{Q} &= \textit{l'ensemble des nombres rationels} \\ &= \textit{l'ensemble des classes d'équivalence de la relation } \mathcal{R} \end{aligned}$$

Example 1.17

$$\begin{aligned} S &= \{(a, b) \ : a, b \in Z_+\} \quad \text{et } (a, b)\mathcal{R}(c, d) \Longleftrightarrow a + d = b + c \\ Z &= \textit{l'ensemble des entiers. Il est caractérisé par l'ensemble des classes} \\ &\quad \textit{d'équivalence de la relation } \mathcal{R} \end{aligned}$$

L'entier -3 est représenté par la classe d'équivalence

$$E((1, 4)) = \{(0, 3), (1, 4), (2, 5), (3, 6), \cdots\}.$$

1.9 Ordre Partiel (Partially Ordered Set)

Definition 1.9 Une relation binaire qui est réflexive, antisymétrique et transitive (sur un ensemble A) définit un *ensemble partiellement ordonné* (A, α), ou un *ordre partiel* α sur A.

Example 1.18

$$A = Z \;\; x\alpha y \Leftrightarrow x \geq y \;\; \textit{est un ordre partiel sur } Z.$$

Example 1.19

$$A = \mathcal{P}(N) \;\; S\alpha T \Leftrightarrow S \subseteq T$$

α *est un ordre partiel (inclusion des ensembles) sur* $\mathcal{P}(N)$.

Example 1.20

$$A = Z_+ \setminus \{0\} \; x\alpha y \Leftrightarrow x|y^3.$$

α *est un ordre partiel (division des entiers positifs).*

Example 1.21 *Etant donnée une fonction* f *définie sur* A*, une fonction partielle est définie par* f *et par* $A' \subseteq A$*. Considérons la famille* $\mathcal{F}$ *de fonctions partielles*

$$f_{A'} \;:\; A \to B \cup \{w\} \textit{est définie par} \;\; f_{A'}(x) = f(x) \quad \textit{si } x \in A'$$

et $f_{A'}(x) = \omega$ *si* $x \in A \backslash A'$.

Deux fonctions $f_{A_1}, f_{A_2} \in \mathcal{F}$ sont telles que $f_{A_1} \alpha f_{A_2} \Leftrightarrow A_1 \subseteq A_2$

c.-à.-d. f_{A_1} est moins définie que f_{A_2} donne un ordre partiel sur $\mathcal{F}$.

Les ordres partiels (Posets) peuvent se représenter par des *diagrammes de HASSE*.

Example 1.22 *Divisibilité des entiers* $\{1,2,\ldots,12\}$.

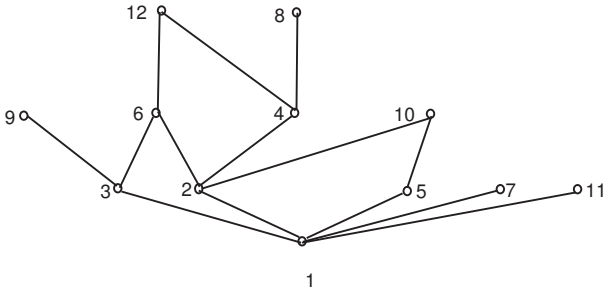


Figure 1.3: (Chaque élément est relié à l'élément le plus "proche" de lui, c'est-à-dire à l'élément pour lequel la relation ne peut pas être découpée par transitivité)

[3] $x|y \equiv x$ divise y

Example 1.23 *Sous-ensembles de* $\{1, 2, 3\}$

$$A\mathcal{R}B \leftrightarrow A \subseteq B \ \forall A, B \subseteq \{1, 2, 3\}$$

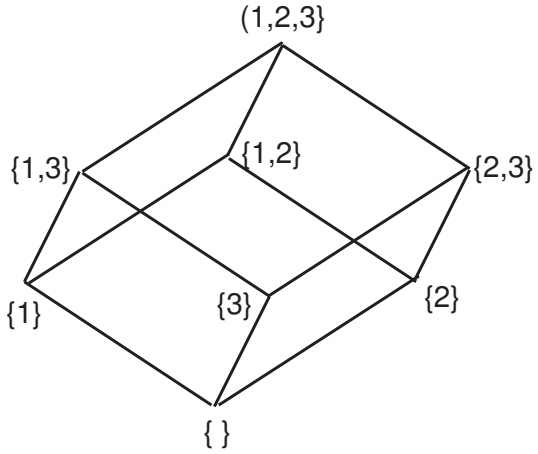


Figure 1.4: Arc $(x, y) \Longleftrightarrow x\mathcal{R}y$ et $\nexists z(\neq x, y)$ tel que $x\mathcal{R}z$ et $z\mathcal{R}y$

1.10 Relation 1-aire

Definition 1.10 Une *relation n-aire* α *sur* $A_1 \times A_2 \times \ldots \times A_n$ est un sous-ensemble

$$\mathcal{R}_\alpha \subseteq A_1 \times A_2 \times \cdots \times A_n \text{ (produit cartésien)}$$

où $(a_1, a_2, \ldots, a_n)\alpha \Leftrightarrow (a_1, a_2, \ldots, a_n) \in \mathcal{R}_\alpha$

Example 1.24 *Relation ternaire sur* $Z_+ \times Z_+ \times Z_+$

$$(x, y, z)\alpha \Longleftrightarrow x^2 + y^2 = z^2$$

1.11 Ensembles finis et infinis

Definition 1.11 Deux ensembles A et B sont *équipotents* (equinumerous) s'il existe une bijection

$$f : A \to B.$$

On peut vérifier facilement que cette relation est une équivalence (car il existera aussi une bijection $f^{-1} : B \to A$ et la composée de bijections est une bijection. Cette relation d'équivalence définit donc une partition de la famille de tous les ensembles.

On peut donc considérer (il s'agit d'une explication et pas d'une définition) un ensemble A comme *fini* s'il est équipotent avec $\{1, \ldots n\}$ pour un certain $n \in N$. Dans ce cas on dira que le *cardinal* de A ($|A|$) est n. Le cardinal d'un ensemble fini est donc le nombre d'éléments qu'il contient.

Un ensemble *infini* est un ensemble qui n'est pas fini. Tous les ensembles infinis ne sont pas équipotents. Un ensemble est *infini dénombrable* (countably infinite) s'il est équipotent avec $\mathbb{N}$. Un ensemble est *dénombrable* s'il est équipotent à une partie de $\mathbb{N}$ (c.à.d. fini ou infini dénombrable).

Pour démontrer qu'un ensemble A est infini dénombrable, le chemin le plus direct est de trouver une bijection entre A et un ensemble B qui *est* infini dénombrable.

Proposition 1.4 *L'union d'un ensemble dénombrable (fini et infini) d'ensembles infinis dénombrables est infini dénombrable.*

Preuve 1. Prenons le cas où l'on prend l'union d'un nombre fini d'ensembles infinis dénombrables. Soit

$$A = \{a_0, a_1, \ldots\} \; ; \; B = \{b_0, b_1, \ldots\} \; ; \; C = \{c_0, c_1, \ldots\}$$
$$A \cup B \cup C = \{a_0, b_0, c_0, a_1, b_1, c_1, \ldots\}$$

On doit trouver une bijection entre $A \cup B \cup C$ et $\mathbb{N}$

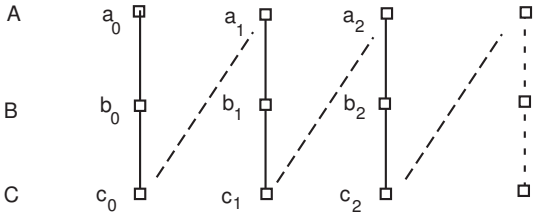


Figure 1.5:

L'énumeration dans cet ordre construit une telle bijection.

2. La même idée peut être utilisée dans le cas de l'union d'une collection infinie dénombrable d'ensembles infinis dénombrables. Donc

$$\cup_{i=0}^{\infty} A_i \text{ avec } A_i = \{a_{i0}, a_{i1}, a_{i2}, a_{i3}, \ldots\}.$$

On peut observer que ceci revient à montrer que $\mathbb{N} \times \mathbb{N} = \{(i,j)\}_{i,j=0,1,\ldots}$ est infini dénombrable.

Dans ce cas, on doit être plus prudent en utilisant la technique ci-dessus. On ne peut plus dans l'énumération visiter un élément de chaque ensemble avant de visiter le 2^{e} élément du premier ensemble (on ne finirait jamais le premier passage).

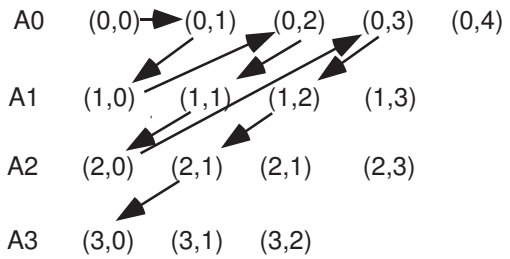


Figure 1.6: Ordre d'enumération pour la construction de la bijection.

Au k^e tour, nous visitons toutes les paires (i, j) avec $i + j = k - 1$ dans l'ordre $\{(k-1, 0), (k-2, 1), \ldots, (0, k-1)\}$. Ceci donne:

$$(0,0), (1,0), (0,1), (2,0), (1,1), (0,2), (3,0), \ldots$$

∎

Chapter 2

Logique des Propositions: une Introduction

2.1 Les Buts

Le but traditionnnel de la logique mathématique a été de fournir des règles rendant le raisonnement précis et correct. Cest à dire qu'on s'intéresse plus à la forme des arguments qu'aux arguments eux-mêmes. Depuis peu, elle est devenue un outil pratique en informatique pour exprimer les buts, ou les buts présumés, des programmes. Ces techniques jouent un rôle central dans l'automatisation de la programmation, ainsi que dans la synthèse, la vérification et le "debugging" des programmes, et dans des langages (tels LISP et PROLOG) qui utilisent des phrases logiques comme programmes.

Dans la suite nous allons étudier la logique des Propositions. Pour la logique des Prédicats la démarche est similaire.

Parce que les langages naturels sont ambigus:

1. il faut un **langage**, composé de **symboles**, et des règles de construction de **phrases** à partir des symboles (les règles de syntaxe).

   Par exemple

   (a) Une ligne correcte de PASCAL ou de FORTRAN;

   (b) Une phrase en français avec la structure

NOM	–	VERBE	–	OBJET
(La banane)	–	mange	–	(le singe)

   (c) $3x^2 = 2x + 1$

2. Il faut donner un **sens** aux phrases du langage: en donnant une **interprétation** aux symboles, nous pouvons donner une **valeur** (Vrai ou Faux) à une phrase. Cette interprétation est la *sémantique*.

   Par exemple:

(c) Sous l'interprétation '$x = 1$', la phrase "$3x^2 = 2x + 1$", est VRAIE.

3. Nous allons distinguer différentes phrases:

   (a) Les phrases qui sont VRAIES sous toutes les interprétations: elles sont des phrases VALIDES, ou des TAUTOLOGIES;

   (b) Les phrases qui sont "SATISFIABLES", c'est à dire, des phrases pour lesquelles il existe une interprétation sous laquelle elles sont VRAIES;

   (c) Les phrases qui sont FAUSSES sous toutes les interprétations: ce sont des CONTRADICTIONS;

   En outre nous considérons l'**implication** et l'**équivalence** des phrases.

4. Maintenant nous pouvons l'utiliser de différents façons:

   (a) Si une phrase est une TAUTOLOGIE, on a un THEOREME.

   (b) Si l'on utilise les "implications", les "équivalences", etc. pour démontrer qu'une phrase est une TAUTOLOGIE, cette démonstration est alors une **preuve logique**.

   (c) Vérifier qu'un program est correcte ressemble fort à démontrer un théorême.

5. La logique des prédicats est beaucoup plus riche. Elle contient des variables, et les unificateurs $(\forall x)$ et $(\exists x)$. Ceci permets des arguments beaucoup plus interessants. Par exemple une direction est de choisir une **classe d'interprétations** et de développer une **théorie spécialisée**: la théorie mathématique des entiers, des ensembles, des graphes par exemple (**Model Theory**).

2.2 Introduction

Prenons quelques exemples.

Même si nous ne savons rien sur le prochain gouvernement, nous savons que la phrase:

$$\left(\begin{array}{c}\text{Le prochain premier ministre sera une femme}\\ \text{ou}\\ \text{le prochain premier ministre ne sera pas une femme.}\end{array}\right)$$

est vraie.

De même pour:

$$\left(\begin{array}{c}\text{La ville d'Ypres a moins de 20.000 habitants}\\ \text{ou}\\ \text{La ville d'Ypres n'a pas moins de 20.000 habitants.}\end{array}\right)$$

Ces deux phrases sont des instances de la phrase abstraite:

$$\left(\begin{array}{c}P\\ \text{ou}\\ \text{non } P\end{array}\right) \text{ qui est notée } (P \vee \neg P)$$

Toute phrase de cette forme est vraie indépendamment de la verité ou non de P.
Nous dirons qu'une phrase abstraite est *Valide (Tautologie)* si elle est vraie indépendamment des valeurs de ses composants.

Valide:	$\left(\begin{array}{c}\text{non } (P \text{ et } (\text{non } P)\\ \text{ou}\\ Q\end{array}\right)$	$\neg(P \wedge \neg P) \vee Q$
Contradiction	$(P \text{ et } (\text{non } P))$	$P \wedge \neg P$
Ni Valide ni une Contradiction:[1]		$P \vee Q, \quad \neg P$ etc.
Equivalence:		
Les deux phrases:	(si P alors Q)	$P \rightarrow Q$
	(si (non Q) alors (non P))	$\neg Q \rightarrow \neg P$

sont équivalentes dans le sens qu'une des phrases est vraie/fausse précisément quand l'autre est vraie/fausse.

Les deux phrases

- *S'il pleut, alors les rues sont mouillées.*
- *Si les rues ne sont pas mouillées, il ne pleut pas*

[1] Il y a des instances vraies et des instances fausses.

sont de la forme $P \to Q$ et $\neg Q \to \neg P$.
La **Logique des propositions** nous permettra de déterminer la validité ou non de beaucoup de phrases abstraites uniquement sur base de leur forme.

Example 2.1 *Soit G un graphe à n sommets, et considérons les trois phrases*

- $\mathcal{F}_1$*: Si G est acyclique, alors G contient au plus $n-1$ arêtes.*
- $\mathcal{F}_2$*: Si G contient au moins n arêtes, G contient un cycle.*
- $\mathcal{F}_3$*: Soit G contient un cycle, soit G contient au plus $n-1$ arêtes.*

Est-ce que ces phrases sont des Tautologies (vraies)?
Doit-on prouver toutes les trois?
En effet, les trois sont logiquement équivalent. Donc il suffit d'en démontrer une.
Décomposons ces phrases.

- *A: G est acyclique.*
- *B: G contient au plus $n-1$ arêtes.*

Maintenant on peut les réécrire

- $\mathcal{F}_1$*: Si A, alors B, ou formellement $A \to B$*
- $\mathcal{F}_2$*: Si pas B, alors pas A ou formellement $\neg B \to \neg A$*
- $\mathcal{F}_3$*: Soit pas A, soit B ou formellement $\neg A \wedge B$.*

Nous allons montrer que les trois phrases sont équivalentes:

$$A \to B \Leftrightarrow \neg B \to \neg A \Leftrightarrow \neg A \wedge B.$$

2.3 Le langage (Syntaxe)

Definition 2.1 Les *phrases* de la logique des propositions sont construites à partir des symboles, appellés **propositions** ou **propositions primitives**

- Les symboles de vérité: **'V'** et **'F'**;
- Les symboles propositionnels: $P, Q, R, P_1, P_2, R_1, R_2 \ldots$

Definition 2.2 Les *phrases* sont construites à partir des propositions par l'application des **connecteurs propositionnels:**

$\neg$	$\wedge$	$\vee$	$\to$	$\leftrightarrow$
non	et	ou	si alors	si et seulement si

La construction des *phrases* se fait selon les règles suivantes:

1. Les propositions sont des phrases;
2. Si $\mathcal{F}$ est une phrase, $\neg\mathcal{F}$ est une phrase **(négation)**
3. Si $\mathcal{F}$ et $\mathcal{G}$ sont des phrases,
   (a) $(\mathcal{F} \wedge \mathcal{G})$ est une phrase **(conjonction)**
   (b) $(\mathcal{F} \vee \mathcal{G})$ est une phrase **(disjonction)**
   (c) $(\mathcal{F} \rightarrow \mathcal{G})$ est une phrase **(implication)**
   (d) $(\mathcal{F} \leftrightarrow \mathcal{G})$ est une phrase **(équivalence)**

Remarque:
$\mathcal{F}$ et $\mathcal{G}$ représentent des phrases. Ces deux symboles ne font pas partie du langage des propositions, mais de notre métalangage.

Example 2.2 $((\neg(P \vee Q)) \leftrightarrow ((\neg P) \wedge (\neg Q)))$ *est une phrase parce que:*

- P *et* Q *sont des phrases;*
- $(P \vee Q)$, $(\neg P)$ *et* $(\neg Q)$ *sont des phrases;*
- $(\neg(P \vee Q))$ *et* $((\neg P) \wedge (\neg Q))$ *sont des phrases;*
- *donc* $((\neg(P \vee Q)) \quad \leftrightarrow \quad ((\neg P) \wedge (\neg Q)))$ *est une phrase.*

Notation
Nous allons laisser tomber les parenthèses quand elles ne sont pas néçessaires: par exemple, les parenthèses extérieures: $\neg P$ pour $(\neg P)$, etc.
À propos des formules propositionelles, on parle souvent de **fbf** (en anglais **wff**) ce qui signifie *formule bien formée* (en anglais, *well-formed formula*).

2.4 Le sens d'une phrase (Sémantique)

Maintenant nous allons associer une valeur de vérité vrai (V) ou faux (F) à chaque phrase (**fbf**).[2]

Definition 2.3 Une **interprétation** ("valuation") J est une affectation d'une valeur de vérité V ou F à un ensemble de symboles propositionnels.

Pour une phrase (**fbf**) $\mathcal{F}$, une interprétation J est une *interprétation pour* $\mathcal{F}$ si J affecte une valeur à chacun des symboles propositionnels de $\mathcal{F}$.

[2] **'V'** et **'F'** sont des "symboles" qui représentent des phrases respectivement vraies ou fauses. V et F son les "valeurs de vérité" qui, associées à un ensemble de symboles propositionnels, déterminent qu'une phrase soit vraie ou fause.

Example 2.3
Soit la phrase

$$\mathcal{F} : P \vee \neg Q$$

Une interprétation possible J_1 pour $\mathcal{F}$ est:

$$\begin{aligned} P &: \mathsf{F} \\ Q &: \mathsf{V} \end{aligned}$$

Une autre interprétation possible J_2 pour $\mathcal{F}$:

$$\begin{aligned} P &: \mathsf{V} \\ Q &: \mathsf{V} \end{aligned}$$

Si on a donné une valeur aux symboles, quelle est la valeur de la phrase?

Definition 2.4 *Règles sémantiques* Soit $\mathcal{E}$ une phrase (**fbf**) et J une interprétation de $\mathcal{E}$, la **valeur** de $\mathcal{E}$ sous J est obtenue par les règles suivantes:

1. La valeur des symboles $P,\ Q,\ R$ dans $\mathcal{E}$ est la même que la valeur affectée par J;
2. La phrase **'V'** est V sous J,
   La phrase **'F'** est F sous J;
3. $\neg\mathcal{F}$ est V si $\mathcal{F}$ est F, et F si $\mathcal{F}$ est V;
4. $\mathcal{F} \wedge \mathcal{G}$ est V si $\mathcal{F}$ et $\mathcal{G}$ sont V, et F sinon;
5. $\mathcal{F} \vee \mathcal{G}$ est V si $\mathcal{F}$ ou $\mathcal{G}$ est V, et F si $\mathcal{F}$ et $\mathcal{G}$ sont F;
6. $\mathcal{F} \rightarrow \mathcal{G}$ est F si $\mathcal{F}$ est V et $\mathcal{G}$ est F, et V sinon;
7. $\mathcal{F} \leftrightarrow \mathcal{G}$ .....

Pour mieux voir, nous pouvons utiliser les **Tables de vérité**

$\mathcal{F}$	$\mathcal{G}$	$\mathcal{F} \wedge \mathcal{G}$	$\mathcal{F} \vee \mathcal{G}$	$\mathcal{F} \rightarrow \mathcal{G}$
V	V	V	V	V
V	F	F	V	F
F	V	F	V	V
F	F	F	F	V

F	$\neg$F
V	F
F	V

Example 2.4
Soit la phrase: $(P \vee \neg Q) \rightarrow Q$; Les interprétations J suivantes sont possibles:

	P	Q	$\neg Q$	$P \vee \neg Q$	$(P \vee \neg Q) \rightarrow Q$
J_1	V	V	F	V	V
J_2	V	F	V	V	F
J_3	F	V	F	F	V
J_4	F	F	V	V	F

2.5 Propriétés des phrases

Definition 2.5 Une phrase (**fbf**) est **valide** (est une tautologie) si elle est vraie pour toute interprétation.

Definition 2.6 Une phrase est **satisfiable** s'il existe une interprétation pour laquelle elle est vraie.

Definition 2.7 Une phrase est une **contradiction** si elle est fausse pour toute interprétation.

Definition 2.8 Une phrase $\mathcal{F}$ **implique** ($\Rightarrow$) une phrase $\mathcal{G}$ si pour toute interprétation J pour $\mathcal{F}$ et $\mathcal{G}$ nous avons que si $\mathcal{F}$ est vraie sous J, $\mathcal{G}$ est aussi vraie sous J.

Definition 2.9 Deux phrases $\mathcal{F}$ et $\mathcal{G}$ sont **équivalentes** ($\Leftrightarrow$) si pour toute interprétation de $\mathcal{F}$ et $\mathcal{G}$, $\mathcal{F}$ a la même valeur de vérité que $\mathcal{G}$.

Definition 2.10 Un ensemble de phrases $\mathcal{F}_1, \ldots, \mathcal{F}_r$ est **consistent** s'il existe une interprétation sous laquelle chaque $\mathcal{F}_i$ est vraie pour $i = 1, \ldots, r$.

Example 2.5 *La phrase P est satisfiable.*

Example 2.6 *La phrase $(P \vee \neg P)$ est valide et satisfiable.*

Example 2.7 *La phrase $(P \wedge \neg P)$ est une contradiction.*

Example 2.8 *La phrase $(P \wedge Q)$ implique la phrase P.*

Example 2.9 *Les phrases $(P \rightarrow Q)$ et $(\neg Q \rightarrow \neg P)$ sont équivalentes.*

Example 2.10 *L'ensemble de phrases P, $P \vee Q$ et $\neg Q$ est consistent.*

Nous pouvons aussi comprendre les définitions autrement:

- $\mathcal{F}$ *implique* $\mathcal{G}$ peut être défini comme: $(\mathcal{F} \rightarrow \mathcal{G})$ est *valide*;
- $\mathcal{F}$ est *équivalente* à $\mathcal{G}$ peut être défini comme: $(\mathcal{F} \leftrightarrow \mathcal{G})$ est *valide*;
- $\mathcal{F}$ est *satisfiable*, est la même chose que $(\neg\mathcal{F})$ n'est pas une *contradiction*.

2.6 Vérification par Tables de Vérité

Comment vérifier qu'une phrase $\mathcal{F}$ est valide, ou que deux phrases $\mathcal{F}$ et $\mathcal{G}$ sont équivalentes?

Il est clair que pour la logique des propositions on peut toujours utiliser les tables de vérite. Il y a donc un algorithme fini pour répondre à ces questions, mais ceci implique beaucoup de travail.

Example 2.11
Preuve que $(P \to Q) \Longleftrightarrow (\neg P \vee Q)$ *ou que* $\Big((P \to Q) \leftrightarrow (\neg P \vee Q)\Big)$ *est valide:*

P	Q	$\neg P$	$P \to Q$	$\neg P \vee Q$	$(P \to Q) \leftrightarrow (\neg P \vee Q)$
V	V	F	V	V	V
V	F	F	F	F	V
F	V	V	V	V	V
F	F	V	V	V	V

A part les équivalences, un autre concept important est celui de l'implication (voir définition 10.5.4). Rappelons que $\mathcal{F} \Rightarrow G$ si $(\mathcal{F} \to G)$ est valide (une tautologie).

Example 2.12

$$P \wedge Q \Rightarrow P$$

Nous montrerons que $(P \wedge Q) \to P$ *est une tautologie.*

P	Q	$P \wedge Q$	$(P \wedge Q) \to P$
V	V	V	V
V	F	F	V
F	V	F	V
F	F	F	V

Example 2.13 *Supposons qu'on a déjà démontré les deux théorèmes suivants:*
TH1: Si G *est un arbre,* G *est acyclique.*
TH2: Si G *a au moins* n *arêtes,* G *contient un cycle.*
Nous voulons démontrer le théorème:
TH: Si G *est un arbre,* G *ne contient pas plus que* $n-1$ *arêtes.*

DEMONSTRATION:
Soit A *la proposition:* G *est un arbre Soit* B *la proposition:* G *est un acyclique*
Soit C *la proposition:* G *contient au moins* n *arêtes.*
Alors par hypothèse:
TH1: $A \to B$
TH2: $C \to \neg B$

$$(A \to B) \wedge (C \to \neg B) \Leftrightarrow (A \to B) \wedge (\neg(\neg B) \to \neg C)$$
$$\Leftrightarrow (A \to B) \wedge (B \to \neg C)$$
$$\Rightarrow (A \to \neg C)$$

Donc on a prouvé:
TH: Si G est un arbre, G ne contient pas plus que $n-1$ arêtes.

■

2.7 Utilisation formelle des equivalences et implications

Le plus important pour un raisonnement logique sont les équivalences et implications suivantes.

Table d'IMPLICATIONS tautologiques

$P \wedge Q \Rightarrow P$	1	
$P \wedge Q \Rightarrow Q$	2	
$P \Rightarrow P \vee Q$	3	
$\neg P \Rightarrow P \rightarrow Q$	4	
$Q \Rightarrow P \rightarrow Q$	5	
$\neg(P \rightarrow Q) \Rightarrow P$	6	
$\neg(P \rightarrow Q) \Rightarrow \neg Q$	7	
$P \wedge (P \rightarrow Q) \Rightarrow Q$	8	(Modus Ponens)
$\neg Q \wedge (P \rightarrow Q) \Rightarrow \neg P$	9	
$\neg P \wedge (P \vee Q) \Rightarrow Q$	10	
$(P \rightarrow Q) \wedge (Q \rightarrow R) \Rightarrow P \rightarrow R$	11	(Enchaînement)
$(P \vee Q) \wedge (P \rightarrow R) \wedge (Q \rightarrow R) \Rightarrow R$	12	

Table des ÉQUIVALENCES

E_1	$\neg\neg P \Leftrightarrow P$	(double négation)
E_2	$P \wedge Q \Leftrightarrow Q \wedge P$	(lois de commutativité)
E_3	$P \vee Q \Leftrightarrow Q \vee P$	
E_4	$(P \wedge Q) \wedge R \Leftrightarrow P \wedge (Q \wedge R)$	(lois d'associativité)
E_5	$(P \vee Q) \vee R \Leftrightarrow P \vee (Q \vee R)$	
E_6	$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$	(lois de distributivité)
E_7	$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$	
E_8	$\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$	(lois De Morgan)
E_9	$\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$	
E_{10}	$P \vee P \Leftrightarrow P$	
E_{11}	$P \wedge P \Leftrightarrow P$	
E_{12}	$R \vee (P \wedge \neg P) \Leftrightarrow R$	
E_{13}	$R \wedge (P \vee \neg P) \Leftrightarrow R$	
E_{14}	$R \vee (P \vee \neg P) \Leftrightarrow T$	
E_{15}	$R \wedge (P \wedge \neg P) \Leftrightarrow F$	
E_{16}	$P \rightarrow Q \Leftrightarrow \neg P \vee Q$	
E_{17}	$\neg(P \rightarrow Q) \Leftrightarrow P \wedge \neg Q$	
E_{18}	$P \rightarrow Q \Leftrightarrow \neg Q \rightarrow \neg P$	
E_{19}	$P \rightarrow (Q \rightarrow R) \Leftrightarrow (P \wedge Q) \rightarrow R$	
E_{20}	$\neg(P \leftrightarrow Q) \Leftrightarrow P \leftrightarrow \neg Q$	
E_{21}	$P \leftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$	
E_{22}	$(P \leftrightarrow Q) \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q)$	

Propriétés importantes des équivalences et implications

1. $P \leftrightarrow Q \Longleftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$
2. Si un formule est équivalente à une tautologie, alors elle est aussi une tautologie.
3. Si un formule est impliquée par une tautologie, alors elle est aussi une tautologie.
4. *Implication* et *Équivalence* sont transitives.
5. Si $A \Rightarrow B$ et $A \Rightarrow C$, alors $A \Rightarrow B \wedge C$.
6. Si $H_1, H_2, \ldots, H_m$ impliquent tous B, alors $(H_1 \wedge H_2 \ldots \wedge H_m) \Rightarrow B$.

Theorem 2.1 *Si $H_1, H_2, \ldots, H_m$ impliquent Q;*
si $P \Rightarrow Q$;
alors $H_1, \ldots, H_m$ impliquent $P \rightarrow Q$

Preuve
De la propriété 6 ci-dessus on déduit que

$$H_1 \wedge H_2 \ldots \wedge H_m \wedge P \Rightarrow Q.$$

Donc

$$H_1 \wedge H_2 \ldots \wedge H_m \wedge P \rightarrow Q$$

est une tautologie.
En utilisant l'équivalence

$$P_1 \rightarrow (P_2 \rightarrow P_3) \Longleftrightarrow (P_1 \wedge P_2) \rightarrow P_3$$

on obtient la tautologie $(H_1 \wedge \ldots \wedge H_m) \rightarrow (P \rightarrow Q)$ ∎

Example 2.14 *Montrer de nouveau que*

$$(A \rightarrow B), (C \rightarrow \neg B) \Rightarrow A \rightarrow \neg C.$$

1.	$A \rightarrow B$			
2.	$C \rightarrow \neg B$			
3.	$\neg(\neg B) \rightarrow \neg C$	$E18$	(2)	*Contrapositive*
4.	$B \rightarrow \neg C$	$E1$	(3)	*Double Négative*
5.	$A \rightarrow \neg C$	$I11$	$(1,4)$	*Enchainement*

Example 2.15 *Transformer en forme symbolique et tester la validité de l'argument suivant en utilisant les règles d'implication et d'équivalence*
a) Si Jean obtient le poste de directeur et il travaille bien, alors il obtient une augmentation.
b) Jean achétera une nouvelle voiture s'il obtient une augmentation.
c) Jean n'a pas acheté une nouvelle voiture

Alors
d) Soit Jean ne travaillait pas bien, soit il n'a pas obtenu le poste de directeur.

Formellement, soit
A: Jean obtient le de directeur
B: Jean travaille bien
C: Jean obtient une augmentation
D: Jean achète une nouvelle voiture

Les hypothèses sont:
1.$(A \wedge B) \rightarrow C$
2.$C \rightarrow D$
3.$\neg D$
La conclusion: $\neg B \vee \neg A$.

L'argument

4.	$\neg D \rightarrow \neg C$	$E18$	(2)
5.	$\neg C$	$I8$	(3), (4)
6.	$\neg C \rightarrow \neg(A \wedge B)$	$E18$	(1)
7.	$\neg(A \wedge B)$	$I8$	(5), (6)
8.	$\neg A \vee \neg B$	$E8$	(7)
9.	$\neg B \vee \neg A$	$E3$	(8)

2.8 Le nombre de propositions distincts

Question 1

Si $\mathcal{F}$ contient n symboles $P_1, \ldots, P_n$, combien d'interprétations différentes y-a-t-il pour $\mathcal{F}$?

Example 2.16
Soit la phrase $P_1 \vee (P_2 \wedge \neg P_1)$. *Nous dessinons la table de vérité:*

P_1	P_2	$\neg P_1$	$P_2 \wedge \neg P$	$P_1 \vee (P_2 \wedge \neg P_1)$
V	V	F	F	V
V	F	F	F	V
F	V	V	V	V
F	F	V	F	F

Pour $n = 2$ *il y a 4 interprétations différentes.*
Pour $n = 3$ *nous aurions 8 interprétations différentes:*

P_1	P_2	P_3	
V	V	V	
V	V	F	
V	F	V	
F	V	V	
V	F	F	
F	V	F	
F	F	V	
F	F	F	

En général il y a 2^n interprétations différentes.

Question 2

Il y a une infinité de phrases qui contiennent n symboles. Combien de classes d'équivalence différentes y-a-t-il?

Example 2.17
Soit une phrase avec un seul symbole ($n = 1$):
il y a 4 phrases possibles: $P,\ \neg P,\ P \vee \neg P,\ P \wedge \neg P$.

	Phrases			
	P	$\neg P$	$P \vee \neg P$	$P \wedge \neg P$
J_1	V	F	V	F
J_2	F	V	V	F

En effet, nous savons qu'en général il y a 2^n interprétations possibles Or, pour chaque interprétation, la valeur de la phrase peut être V ou F. Il y a donc 2^{2^n} phrases différentes.

2.9 Une algèbre booléenne: les propriétés

La structure associée à notre logique de propositions

$$(\wedge, \vee, \mathsf{V}, \mathsf{F}, \neg)$$

s'appelle *une algèbre booléenne*, parce que les cinq lois définissent une telle algèbre.

1. $\left.\begin{array}{lcl} A \wedge B & \Longleftrightarrow & B \wedge A \\ A \vee B & \Longleftrightarrow & B \vee A \end{array}\right\}$ Commutativité

2. $\left.\begin{array}{lcl} A \wedge (B \wedge C) & \Longleftrightarrow & (A \wedge B) \wedge C \\ A \vee (B \vee C) & \Longleftrightarrow & (A \vee B) \vee C \end{array}\right\}$ Associativité

3.
$$\left.\begin{array}{lcl} A \wedge \mathsf{V} & \iff & A \\ A \wedge \mathsf{F} & \iff & \mathsf{F} \\ A \vee \mathsf{V} & \iff & \mathsf{V} \\ A \vee \mathsf{F} & \iff & A \end{array}\right\} \quad \text{Propiétés de } \vee, \wedge$$

4.
$$\left.\begin{array}{lcl} A \wedge \neg A & \iff & \mathsf{F} \\ A \vee \neg A & \iff & \mathsf{V} \end{array}\right\} \quad \text{Négation}$$

5.
$$\left.\begin{array}{lcl} A \wedge (B \vee C) & \iff & (A \wedge B) \vee (A \wedge C) \\ A \vee (B \wedge C) & \iff & (A \vee B) \wedge (A \vee C) \end{array}\right\} \text{ Distributivité}$$

Chacune peut être vérifiée par une table de vérité.
Une loi très utile est la loi de *De Morgan*:
$$\begin{array}{lcl} \neg(A \wedge B) & \iff & (\neg A) \vee (\neg B) \\ \neg(A \vee B) & \iff & (\neg A) \wedge (\neg B) \end{array}$$

Chapter 3

Techniques de Démonstration

Ici nous allons essayer de rassembler différentes techniques de démonstration. En particulier, il y a les arguments basés sur les "règles d'inférence" logiques; mais il y a aussi d'autres techniques "mathématiques" associes aux entiers.

3.1 Inférence Logique

Dans le chapitre précédent, nous avons vu les "règles d'inférence" qui permettent de partir d'un ensemble d'hypothèses et d'en tirer une conséquence logique.

Règles d'inférence

$p \to (p \vee q)$	Addition
$(p \wedge q) \to p$	Simplification
$[p \wedge (p \to q)] \to q$	Modus ponens
$[\neg q \wedge (p \to q)] \to \neg p$	Modus tollens
$[(p \to q) \wedge (q \to r)] \to (p \to r)$	Syllogisme par hypothèse
$[(p \vee q) \wedge \neg p] \to q]$	Syllogisme disjonctif

et il y en a d'autres avec des prédicats

$$(\forall\ x)P(x) \to (\forall\ x)\big(P(x) \vee Q(x)\big)$$
$$(\forall\ x)P(x) \to P(a)$$
$$P(a) \wedge ((\forall\ x)\big(P(x) \to Q(x)\big) \to Q(a)$$
$$(\exists\ x)P(x) \wedge (\forall\ x)\big(P(x) \to Q(x)\big) \to (\exists\ x)Q(x),\ \text{etc.}$$

3.2 Approches Directe, Indirecte, par l'Absurde

3.2.1 Une démonstration directe

Theorem 3.1 *Si n est impair, alors n^2 est impair.*

Ceci est de la forme $P \rightarrow Q$, où
P: "l'entier n est impair"
Q: "l'entier n^2 est impair"

Démonstration. Dans l'approche *directe*, on prend l'hypothèse que P est vraie et on veut montrer que Q doit être vraie.
Donc n est impair. Ceci implique (définition d'impair) que

$$n = 2k + 1, \text{ où } k \in Z.$$

Alors

$$\begin{aligned} n^2 &= (2k+1)(2k+1) = 4k^2 + 4k + 1 = 2(2k^2 + 2k) + 1 \\ &= 2t + 1 \quad \text{où } t = 2k^2 + 2k \text{ est un entier} \end{aligned}$$

Conclusion: $2t + 1$ est impair. Donc n^2 est impair. ∎

3.2.2 Une démonstration indirecte

Dans une démonstration *indirecte* de la validité de $P \rightarrow Q$, on montre la validité de $\neg Q \rightarrow \neg P$.

Theorem 3.2 *Si $3n + 2$ est impair, alors n est impair.*

Ceci est de la forme $P \rightarrow Q$ où
P: "$3n + 2$ est impair"
Q: "n est impair"

Démonstration. Si n est pair, $n = 2k$.
Alors

$$3n + 2 = 6k + 2 = 2(3k + 1) \text{ est pair}$$

.
On a démontré que $\neq Q \Rightarrow \neg P$. ∎

3.2.3 Preuve par l'absurde

Dans une démonstration de la validité de P, on va montrer que $\neg P \rightarrow$ "F", c'est-à-dire $\neg P$ est toujours faux.

Theorem 3.3 *$\sqrt{2}$ est irrationnel.*

Démonstration. Si $\sqrt{2}$ est rationnel, on peut écrire

$$\sqrt{2} = \frac{p}{q}, \text{ où } p, q \text{ n'ont aucun facteur en commun}$$
$$\text{Alors } p^2 = 2q^2 \Rightarrow p^2 \text{ pair} \Rightarrow p \text{ pair} \Rightarrow p = 2k$$

Maintenant

$$p^2 = 4k^2 = 2q^2 \qquad \Rightarrow q^2 = 2k^2$$
$$\Rightarrow q^2 \text{ pair} \Rightarrow q \text{ pair} \Rightarrow q = 2e.$$

Finalement on a montré que p et q sont divisibles par 2, une contradiction à l'hypothèse qu'ils n'ont aucun facteur en commun. ∎

3.3 Preuves Constructives et Non Constructives

3.3.1 Preuve d'existence constructive $(\forall\ n)P(n)$

Theorem 3.4 *Pour tout entier positif n, il existe une suite de n entiers consécutifs*

$x_n + 1, \cdots, x_n + n$ *qui sont tous composés.*

Démonstration.
Prenons

$$x_n = (n+1)! + 1.$$

et considèrons les entiers

$$x_n + 1, \cdots x_n + n$$

ou autrement dit

$$(n+1)! + (i+1) \text{ pour } i = 1, \cdots, n.$$

On voit que $i + 1$ divise $(n+1)! + (i+1)\ \forall\ i$.
Donc les n nombres consécutifs sont composés. ∎
Ici on a trouvé explicitement la valeur de x_n qui apparaît dans le théorème.
Donc la preuve ets *constructive*.

3.3.2 Preuve d'existence non constructive

Theorem 3.5 $\forall\ n, \exists$ *un nombre premier* $> n$.

Démonstration. Considérons l'entier $n! + 1$.
Soit $z = n! + 1$ est un nombre premier. Alors $z > n$ et z est le nombre premier requis.
Soit z est composé. Alors $z = pq$ où p est un nombre premier > 1. Notons que pour chaque $i = 2, \cdots, n$, on peut écrire

$$z = i \cdot k_i + 1 \text{ où } k_i = \frac{n!}{i} \text{ est un entier.}$$

Donc i ne divise pas z pour $i = 2, \cdots, n$. Mais p divise z, et donc $p > n$ est le nombre premier requis. ∎
Ici on a juste démontré le résultat, mais on ne connaît pas l'entier p.
Donc la preuve est *nonconstructive*.

3.4 Les Entiers et le Principe du Bon Ordre

Nous connaissons toutes les règles élémentaires de calcul et les propriétés de base des entiers. Il y a un axiome fondamental dont nous avons besoin.

Principe du Bon Ordre *(Well-Ordered). Tout ensemble non vide d'entiers non négatifs contient un plus petit élément.*
Logiquement:

$$S \subseteq Z^+ \rightarrow (\exists\ x)(\forall\ y)\big((y \in S \wedge y \neq x) \rightarrow x < y\big).$$

Mathématiquement:

$$\text{Si } S \subseteq Z^+, \text{ alors } \exists\ x \in S \text{ tel que } x < y\ \forall\ y \in S \setminus \{x\}.$$

Exemple. Soit a et d deux entiers. Alors il existe des entiers q et r tel que $a = qd + r$ avec $0 \leq r < d$.

Démonstration. Soit S l'ensemble $\{a - dq\ \forall\ q \in Z : a - dq \geq 0\}$.
S est un ensemble d'entiers non négatifs.
Donc il existe un plus petit entier $r = a - dq_0 \in S$ avec $r \geq 0$.
Supposons (par l'absurde) que $r \geq d$.
Alors $r - d = a - d(q_0 + 1) \in S$, et r n'est pas le plus petit entier dans S.
Contradiction.
Donc $r < d$. ∎

3.5 Principe d'Induction Mathématique

De nombreux théorèmes sont de la forme:

$$\text{“}P(n) \text{ est vraie pour tout entier positif } n\text{”} \quad \text{ou} \quad (\forall\ n)P(n).$$

La technique d'induction est une technique de démonstration pour des théorèmes de ce type.

3.5.1 Principe d'induction

1. *Pas initial:* Vérifier que la proposition $P(n)$ est vraie pour $n = 1$ (ou 0).
2. *Hypothèse d'induction:* Supposons $P(k)$ est vraie pour $k \in Z_+$.
3. *Pas inductif:* On prouve que $P(k + 1)$ est vraie (en utilisant l'hypothèse d'induction).

Exemple 1: Prouvons que $1 + 2 + \cdots + n = \frac{n(n+1)}{2}$

1. $n = 1 : 1 = \frac{1(1+1)}{2}$. Donc $P(1)$ est vraie.

2. Hypothèse: $P(k)$ est vraie, c'est-à-dire $1 + 2 + \cdots + k = \frac{k(k+1)}{2}$

3. $$\begin{aligned} 1 + 2 + \cdots + k + k + 1 &= (1 + 2 + \cdots + k) + (k + 1) \\ &= \frac{k(k+1)}{2} + (k + 1) \text{ utilisant l'hypothèse inductive} \\ &= (k + 1)\,(\tfrac{k}{2} + 1) = \frac{(k+1)(k+2)}{2} \end{aligned}$$

   Donc $P(k + 1)$ est vraie. ∎

Exemple 2: Prouvons que pour tout ensemble fini A : $|\mathcal{P}(A)| = 2^{|A|}$, où $\mathcal{P}(A)$ = l'ensemble des sous-ensembles de A.
Soit $Q(n)$ la proposition: Si $|A| = n$, $|\mathcal{P}(A)| = 2^n$.

1. $n = 1 : A = \{a\}$. $\mathcal{P}(A) = \{\phi, \{a\}\}$.
   Donc $|\mathcal{P}(A)| = 2 = 2^1$. ($Q(1)$ est vraie)

2. Hypothèse: Si $|A| = k$, $|\mathcal{P}(A)| = 2^k$ ($Q(k)$ est vraie)

3. Soit $|A| = k + 1$. Mettons $B = A \setminus \{a\}$ où $a \in A$.
   Alors $|B| = k$ et $|\mathcal{P}(B)| = 2^{|B|} = 2^k$ par l'hypothèse inductive.

   $$\begin{aligned} \mathcal{P}(A) &= \{C : C \subseteq B\} \cup \{C \cup \{a\} : C \subseteq B\} \\ &= \{C : C \in \mathcal{P}(B)\} \cup \{C \cup \{a\} : C \in \mathcal{P}(B)\}. \end{aligned}$$

   Donc $|\mathcal{P}(A)| = 2^k + 2^k = 2^{k+1}$, et la proposition est vraie pour $k + 1$ ($Q(k + 1)$ est vraie).

∎

N.B. C'est aussi vrai pour $n = 0$. Si $A = \emptyset$, par convention $|A| = 0$. Aussi $\mathcal{P}(\emptyset) = \{\emptyset\}$ et donc $|\mathcal{P}(\emptyset)| = 1 = 2^0$.

Remarque. On peut montrer facilement que le principe d'induction est une technique de démonstration valide en utilisant le principe du bon ordre.

Exercice. Echiquier

Principe généralisé d'induction

1. $P(1)$ est vraie

2. Hypothèse inductive: $P(1), P(2), \cdots, P(k)$ sont vraies

3. On montre que $P(k + 1)$ est vraie.

Exemple. Si n est un entier > 1, n peut s'écrire comme le produit de nombres premiers.

Démonstration.

1. $P(2)$ est vrai. $2 = 1 \cdot 2$. 1 et 2 sont des nombres premiers.
2. Hypothèse: $P(1), \cdots, P(k)$ sont vraies.
3. Considérons $k+1$. Soit $k+1$ est premier, soit il ne l'est pas.
   Si $(k+1)$ est premier, $(k+1) = 1 \cdot (k+1)$ et $P(k+1)$ est vrai.
   Si $(k+1)$ n'est pas premier, $k+1 = pq$ où p, q sont deux entiers avec $1 < p, q < k+1$. Par l'hypothèse $P(p)$ et $P(q)$ sont vraies. Alors $p = p_1 p_2 \cdots p_r$ où p_i sont des premiers; $q = q_1 q_2 \cdots q_s$ où q_j sont des premiers. Donc $k+1 = pq = p_1 \cdots p_r q_1 \cdots q_s$ est un produit de nombres premiers, et $P(k+1)$ est vraie.

∎

3.5.2 Principe des Tiroirs (Pigeonhole Principle)

Le principe

Si A et B sont des ensembles finis non vides tels que $|A| > |B|$, alors il n'existe pas de fonction injective: $A \to B$ (c'est-à-dire que si on essaie de remplir les éléments (casiers) de B avec les éléments de A (pigeons), on devra tt ou tard mettre 2 pigeons dans le même casier).

Preuve du principe

Preuve par induction sur $|B|$.
Soit $B = \{b\}$.

1. $|B| = 1$ et $|A| > 1$
   $f : A \to B \quad \Rightarrow \quad \exists\ a_1, a_2 \in A$ avec $a_1 \neq a_2$ et $f(a_1) = f(a_2) = b$
   $\Rightarrow$ f n'est pas injective. Donc $P(1)$ est vraie.
2. Vraie pour $|A| > |B|$ et $|B| \leq k$
3. Soient $f : A \to B$ (où $|A| > |B| = k+1 \geq 2$) et $b \in B$.
   Si $|f^{-1}(b)| \geq 2 \to f$ n'est pas injective.
   Si $|f^{-1}(b)| \leq 1$, alors on considère la fonction

   $$g : A \setminus f^{-1}(b) \to B \setminus \{b\}$$

   avec $g(a) = f(a)\ \forall\ a \in A \setminus f^{-1}(b)$.
   Mais ici, $|B \setminus \{b\}| = k$ et $|A \setminus f^{-1}(b)| \geq |A| - 1 > |B \setminus \{b\}|$.
   Donc on peut appliquer l'hypothèse inductive avec ensembles $A \setminus f^{-1}(b)$ et $B \setminus \{b\}$.
   Par hypothèse g n'est pas injective, ce qui implique que f n'est pas inductive.

∎

Ce principe extrêmement simple est utilisé dans un nombre surprenant de preuves.

Exercice. Combien de personnes faut-il dans une pièce pour être sûr qu'il y en aie au moins deux qui aient leur anniversaire le même jour de la semaine en 2002 ?
Dans cette classe il y a au moins x qui ont leur anniversaire le même jour de la semaine en 2002 ? Quelle est la valeur de x ?

Proposition 3.6 *Soit B un ensemble fini. Considérerons une relation binaire $R_\alpha \subseteq B \times B$. Si chaque élement $b \in B$ est en relation avec un autre élément $v(b)$, c'est-à-dire $(b, v(b)) \in R_\alpha$, alors il existe une suite $b_1, \cdots, b_r$ d'éléments distincts de B telle que*

$$(b_1, b_2), (b_2, b_3), \cdots, (b_{r+1}, b_r), (b_r, b_1)\} \subseteq R_\alpha \subseteq B \times B.$$

Démonstration. Représenter la relation par un graphe dirigé où (b, b') est un arc ssi $(b, b') \in R_\alpha$ avec $n = |B|$ sommets.
Choisir arbitrairement un sommet b_1.
Considérer la suite d'arcs:

$$(b_1, b_2), (b_2, b_3), \cdots, (b_n, b_{n+1}).$$

Par le principe des tiroirs, $b_1, \cdots, b_{n+1}$ ne sont pas tous distincts. Soit b_s le premier qui a déjà apparu, avec $b_s = b_r$ où $r < s$. La suite $(b_s, b_{s+1}), \cdots, (b_{r-1}, b_s)$, est une des suites recherchées. ∎

3.5.3 Principe de diagonalisation

Principe

Soit α une relation binaire sur l'ensemble A.
Soit D le complément de la diagonale $D =\equiv \{a \in A : (a, a) \notin R_\alpha\}$.
Soit $Q_a = \{b \in A : (a, b) \in R_\alpha\}$.
Alors D est distinct de Q_a pour tout $a \in A$

3.5.4 Preuve

Supposons (par l'absurde) que $D = Q_a$ pour $a \in A$

$$\left\{ \begin{array}{lcll} \text{si } a \in Q_a & \Rightarrow & (a, a) \in R_\alpha & \Rightarrow \quad a \notin D \\ \text{si } a \notin Q_a & \Rightarrow & (a, a) \in R_\alpha & \Rightarrow \quad a \in D \end{array} \right\} \text{ d'où contradiction}$$

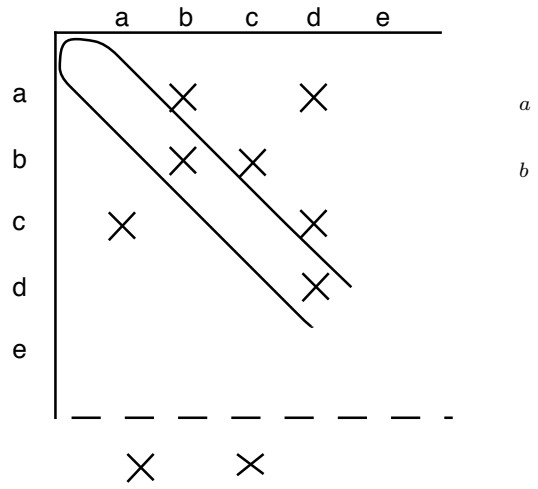


Figure 3.1: Le complément de la diagonale est différent de chaque ligne

■

Le principe de diagonalisation reste vrai pour des ensembles infinis car D diffère toujours de Q_a en ce qui concerne l'élément diagonal a.

Exemple. L'ensemble $\mathcal{P}(\mathbb{N})$ n'est pas dénombrable.

Preuve (Cantor). SI $\mathcal{P}(\mathbb{N})$ était dénombrable, il existerait une bijection $f : \mathbb{N} \to \mathcal{P}(\mathbb{N})$, c'est-à-dire $\mathcal{P}(\mathbb{N}) = \{S_0, S_1, S_2, \cdots\}$.
Soit $D = \{n \in \mathbb{N} : n \notin S_n\}$.
D est un ensemble d'entiers naturels et donc $D = S_k$ pour un certain k.

$\left\{ \begin{array}{l} \text{Si } k \in S_k, \text{ alors } k \notin D \\ \text{Si } k \notin S_k, \text{ alors } k \in D \end{array} \right\}$. On conclut que $k \in S_k \setminus D \cup D \setminus S_k$, et donc $S_k \neq D$. ▌

■

Chapter 4

Dénombrement

4.1 Unions Disjointes et Produits Cartésiens

Proposition 4.1 *Si les A_i sont disjoints, $|\cup_{i=1}^n A_i| = \sum_{i=1}^n |A_i|$.*

Proposition 4.2 (Produit Cartésien) $|A_1 \times A_2 \times \cdots \times A_n| = \prod_{i=1}^n |A_i|$.

Example 4.1 *On doit choisir un académique ou un étudiant pour représenter INMA au sein d'un comité universitaire.*

De combien de façons différentes peut-on sélectionner ce représentant s'il y a 12 académiques et 78 étudiants.
Il y a 12+78 =90 façons possibles.

Maintenant on doit choisir un académique et un étudiant.
Alors il y a 12×78 *façons possibles.*

Example 4.2

i) Combien de fonctions différentes $A \to B$ y a-t-il si $|A| = m$ et $|B| = n$?

$$|f(a_1), f(a_2), \cdots f(a_m)| = |B \times B \times \cdots \times B| = |B|^m = n^m.$$

ii) Quel est le # de sousensembles d'un ensemble A avec $|A| = n$?
Représenter $S \subseteq A$ par $\chi^S = (\chi_1^S, \cdots, \chi_n^S)$ avec $\chi_i^S = 1$ si $i \in S$, $\chi_i^S = 0$ si $i \notin S$.

$$\mathcal{P}(A) = |\underbrace{\{0,1\} \times \{0,1\} \times \cdots \times \{0,1\}}_{n \text{ fois}}| = 2^n$$

iii) Combien de fonctions injectives $A \to B$ y a-t-il si $|A| = m$, $|B| = n$?

$$\begin{array}{ll} \textit{Si } m > n, & \textit{0 (il existe } a_i, a_j \in A \textit{ avec } f(a_i) = f(a_j). \\ \textit{Si } m \leq n, & \textit{n possibilités pour } f(a_1) \\ & n-1 \textit{ possibilités pour } f(a_2) \\ & \qquad \textit{etc.} \end{array}$$

$$n(n-1)\cdots(n-m+1) = \frac{n!}{(n-m)!}$$

Example 4.3 *Mots de passe de 3 à 4 caractères où chaque caractère est une lettre majuscule ou un chiffre. Chaque mot de passe contient au moins un chiffre. Combien de mots de passe distincts y-a-t-il ?*

Soit S*: le nombre de mots de passe distincts.*
Soit P_k*: le nombre de mots de longueur k avec majuscules ou chiffres*
Q_k*: le nombre de mots de longueur k avec majuscules*
R_k*: le nombre de mots de longuer k avec majuscules ou chiffres et au moins un chiffre.*

Par la Proposition 1.1, on a $S = R_3 + R_4$*, et* $P_k = Q_k + R_k$*. Par la Proposition 1.2,* $|P_k| = (36)^k$ *et* $|Q_k| = (26)^k$*.*
Alors $S = (36)^3 - (26)^3 + (36)^4 - (26)^4$*.*

4.2 Permutations et Combinaisons

Soit N un ensemble fini avec n éléments.

Un *ensemble ordonné* de r éléments de l'ensemble est une *r-permutation.*
$P(n,r) \equiv$ le # de r-permutations d'un ensemble de taille n.

Theorem 4.3 $P(n,r) = n(n-1)\cdots(n-r+1)$.

Une *r-combinaison* est une sélection non-ordonnée de r éléments de l'ensemble.
Soit $C(n,r)$ le # de r-combinaisons d'un ensemble de taille n.

Theorem 4.4 $C(n,r) = \dfrac{n!}{r!(n-r)!}$

Preuve: $P(n,r) = C(n,r) \times P(r,r)$.

Corollary 4.1 *Si* $r \leq n$, $C(n,r) = C(n,n-r)$.

Notation: $C(n,r) = \left(\begin{array}{c} n \\ r \end{array} \right) = {}^nC_r$ appelé coefficient binomial.

Example 4.4 *Il y a 8 coureurs dans une course. Le premier va recevoir une medaille d'or, le deuxième reçoit une medaille d'argent, et le troisième la médaille de bronze. De combien de façons les médailles peuvent elles être distribuées?*

Réponse: $8 \times 7 \times 6 = P(8,3)$.

Example 4.5 *De combien de façons peut-on choisir les 5 joueurs dans une équipe de "Génies en Herbe" entre les 12 élèves candidats?*

$C(12,5) = \frac{12!}{5!7!} = \frac{12\times11\times10\times9\times8}{5\times4\times3\times2\times1} = 792.$

Theorem 4.5 (Identité de Pascal)

$$C(n+1,k) = C(n,k-1) + C(n,k)$$

Preuve: Soit $|T| = n+1$, $a \in T$ et $S = T \setminus \{a\}$.

$$\begin{array}{rcl} C(n+1,k) & = & \#\text{ de combinaisons de } k \text{ dans } T \\ & = & \#\text{ de combinaisons de } k \text{ dans } T \text{ qui contiennent } a \\ & + & \#\text{ de combinaisons de } k \text{ dans } T \text{ qui ne contiennent pas } a \\ & = & \#\text{ de combinaisons de } k-1 \text{ dans } T \setminus \{a\} \\ & + & \#\text{ de combinaisons de } k \text{ dans } T \setminus \{a\} \\ & = & {}^nC_{k-1} + {}^nC_k \end{array}$$

Le triangle de Pascal

$$\begin{array}{lccccccccc} & & & & & k & & & & \\ n=0 & & & & & 1 & & & & \\ n=1 & & & & 1 & & 1 & & & \\ n=2 & & & 1 & & 2 & & 1 & & \\ n=3 & & 1 & & 3 & & 3 & & 1 & \\ n=4 & 1 & & 4 & & 6 & & 4 & & 1 \\ & & & & \searrow & & \swarrow & & & \\ & & & & & 10 & & & & \end{array}$$

$${}^5C_2 = {}^4C_1 + {}^4C_2$$

Theorem 4.6 $\displaystyle\sum_{k=0}^{n} {}^nC_k = 2^n$

Preuve: Le $\#$ de sousensembles de $N = \{1, \cdots, n\}$.

$= \sum_{k=0}^{n} (\#$ de sousensembles de taille $k)$

$= \sum_{k=0}^{n} {}^nC_k$

Theorem 4.7 (Identité de Vandermonde)

$${}^{m+n}C_r = \sum_{k=0}^{r} {}^mC_{r-k}\ {}^nC_k$$

Preuve: Soit $|A| = m$, $|B| = n$.

le # de sousensembles de $A \cup B$ de taille r

$=$ $\sum_k$(le # de sousensembles de $A \cup B$ de taille r avec k élements dans B)

$=$ $\sum_k \#$(de sousensembles de A de taille $r-k$ et de sousensembles de B de taille k)

$=$ $\sum_{k=0}^{r} {}^mC_{r-k}C_k^n$ (par Proposition 4.2).

4.3 Théorème du Binome

Theorem 4.8

$$(x+y)^n = \sum_{j=0}^{n} {}^nC_j x^{n-j}y^j$$

$$= {}^nC_0x^n + {}^nC_1x^{n-1}y + \cdots + {}^nC_kx^{n-k}y^k + \cdots + {}^nC_{n-1}xy^{n-1} + C_ny^n$$

Preuve: $\underbrace{(x+y)(x+y)\cdots(x+y)}_{n \text{ termes}}$

Combien de fois va-t-on obtenir le produit $x^{n-k}y^k$? Pour obtenir y^k il faut choisir k fois y. # de fois qu'on va l'obtenir est ${}^nC_k$. ∎

Preuve (par induction)

$$P(1) : (x+y)^1 = x + y = {}^1C_0x^1y^0 + {}^1C_1x^0y^1 \qquad \text{vraie.}$$

Si $P^{(n-1)}$ est vraie

$$(x+y)^{n-1} = \sum_{j=0}^{n-1} {}^{(n-1)}C_j x^{n-1-j}y^j$$

Alors

$$(x+y)^n = (x+y)^{n-1}(x+y) = \sum_{j=0}^{n-1} {}^{n-1}C_j x^{n-1-j}y^j(x+y)$$

$$= \sum_{j=0}^{n} x^{n-j}y^j({}^{n-1}C_j + {}^{n-1}C_{j-1}) = \sum_{j=0}^{n} x^{n-j}y^j \, {}^nC_j$$

∎

Example 4.6 $\sum_{k=0}^{n}(-1)^k \, {}^nC_k = 0$.

Mettre $y = 1, x = -1$ dans le Théorème du Binôme.

Example 4.7 *Une autre preuve que* $\sum_{k=0}^{n} C(n,k) = 2^n$

Mettre $x = y = 1$ *dans le Théorème du Binôme.*

Example 4.8

$$\begin{aligned}(x+y)^4 &= C(4,0)x^4 + C(4,1)x^3y^1 + C(4,2)x^2y^2 + C(4,3)x^1y^3 + C(4,4)y^4 \\ &= x^4 + 4x^3y^1 + 6x^2y^2 + 4x^1y^3 + y^4\end{aligned}$$

Theorem 4.9 *Le nombre de différentes permutations de* n *objets, où il y a* n_k *objets identiques de type* k *est*

$$\frac{n!}{n_1!n_2!\cdots n_k!}$$

Preuve: On peut placer les n_1 objets dans les n dans ${}^{n}C_{n_1}$ positions; et il reste $n-n_1$ positions. On peut placer les n_2 dans ${}^{n-n_1}C_{n_2}$ positions; et il reste $n-n_1-n_2$ positions, etc.

$$\begin{aligned}& {}^{n}C_{n_1}\ {}^{n-n_1}C_{n_2}\cdots \\ = \quad & \frac{n!}{n_1!(n-n_1)!}\frac{(n-n_1)!}{n_2!(n-n_1-n_2)!}\cdots\frac{(n-\sum_{i=1}^{k-2} n_i)!}{n_{k-1}!(n-\sum_{i=1}^{k-1} n_i)!} = \frac{n!}{\prod_{i=1}^{k} n_i!}\end{aligned}$$

Example 4.9 *Il y a une course avec 5 participants, 2 Kenyans, 2 Ethiopiens et un Belge. Pour donner le résultat, on annonce seulement pour chaque position 1ere,...,5e la nationalité de l'athlète. Combien de résultats sont possibles?*
Réponse*:* $\frac{5!}{2!2!1!} = 30$.

Example 4.10 *De combien de façons peut-on distribuer 5 cartes à chacun des 4 joueurs, dans un jeu de 52 cartes.*
Réponse*: La question est différente, mais la réponse est similaire. Joueur 1 obtient* ${}^{5}2C_5$ *possibilités, Joueur 2* ${}^{4}7C_5$*, etc.*
$\frac{52\times51\times\ldots\times33}{5!5!5!5!} = \frac{52!}{5!\ 5!\ 5!\ 5!\ 32!}$

4.4 Permutations et Combinaisons avec Remise

Une *permutation avec remise* est un ensemble ordonné de r objets avec répétitions choisi dans n objets.

Example 4.11 $n = 5$, $r = 2$, $\{a,b,c,d,e\}$. *Il y a 25 possibilités*

aa	ab	ac	ad	ae
ba	bb	bc	bd	be
ca	cb	cc	cd	ce
da	db	dc	dd	de
ea	eb	ec	ed	ee

Theorem 4.10 $f_{pr}(n,r) = n^r$.

Une *combinaison avec remise* est un ensemble non-ordonné de r objets avec répétitions choisi dans n objets.
Soit x_i le nombre de fois l'objet i est choisi. Alors une autre façon de décrire une combinaison avec remise est

$$x_1 + \ldots x_n = r, x \in Z_+^n.$$

Example 4.12 $n = 5$, $\{a,b,c,d,e\}$ $r = 2$. *Il y a 15 possibilités:*

$$aa,\ bb,\ cc,\ dd,\ ee,\ ab,\ ac,\ ad,\ ae,\ bc,\ bd,\ be,\ cd,\ ce,\ de.$$

Les 15 solutions correspondantes de

$$x_1 + x_2 + x_3 + x_4 + x_5 = 2$$

sont $(2,0,0,0,0), (0,2,0,0,0), \ldots, (0,0,0,1,1)$.

Pour nous aider à les compter on va maintenant réécrire quelques solutions
aa devient $aa||||$ *qui devient* 12,
cd devient $||c|d|$ *devient* 35,
be devient $|b|||e$ *devient* 26, *etc.*
Alors le combinaison pq indique qu'il y a un élément en position p et un autre en position q. Maintenant à quel combinaison avec remise corréspond la paire 14*?*

Pour écrire une combinaison avec remise, il faut $n-1$ séperateurs $|$ entre les n éléments disponibles. Ensuite il faut placer r objets entre ces séperateurs, et leur position détermine l'objet. Si k objets sont placés entre le séparateur $i-1$ et i, la combinaison avec remise contient k fois le i^{eme} objet. Finalement il faut choisir r combinaisons dans les $n+r-1$ positions.

Theorem 4.11 $f_{cr}(n,r) = {}^{n+r-1}C_r$.

Example 4.13 *Quel est le nombre de solutions de*

$$x_1 + x_2 + x_3 = 11, \qquad x \in Z_+^3 \ ?$$

Il faut choisir un ensemble de 11 éléments dans 3 types avec répétition.

$$f_{cr}(3,11) = {}^{13}C_{11} = 78.$$

4.5 Principe Inclusion-Exclusion

Observons que

$$A_1 \cup A_2 = (A_1 \setminus A_2) \cup (A_1 \cap A_2) \cup (A_2 \setminus A_1).$$

Parce que les trois ensembles à droite sont disjoints,

$$|A_1 \cup A_2| = (|A_1| - |A_1 \cap A_2|) + |A_1 \cap A_2| + (|A_2| - |A_1 \cap A_2|) = |A_1| + |A_2| - |A_1 \cap A_2|.$$

Généralisant, on obtient

$$|A_1 \cup A_2 \cdots \cup A_k| = \sum_i |A_i| - \sum_{i,j} |A_i \cap A_j| + \sum_{i,j,k} |A_i \cap A_j \cap A_k| \cdots$$

Theorem 4.12 $|A_1 \cup \cdots \cup A_n| = \sum_{r=1}^{n} |A_{i_1} \cap A_{i_2} \cap \cdots \cap A_{i_r}|(-1)^{r+1}$

Preuve: Supposant que $a \in A_1 \cup A_2 \cup \cdots A_k$ est un membre de r des ensembles. Alors il est compté à droite

$$\begin{aligned} &{}^rC_1 - {}^rC_2 + \cdots + (-1)^{r+1} {}^rC_r \\ &= {}^rC_0 = 1 \text{ fois} \end{aligned}$$

∎

Autre forme: Soit N l'ensemble universel et $\bar{A}_i = N \setminus A_i$.

$$\bar{A}_1 \cap \bar{A}_2 \cap \ldots \cap \bar{A}_n = N \setminus (A_1 \cup \ldots \cup A_n)$$
$$\text{Alors } |\bar{A}_1 \cap \bar{A}_2 \cap \ldots \cap \bar{A}_n| = |N| - |A_1 \cup \ldots \cup A_n|$$

Example 4.14 *Quel est le nombre de solutions de*

$$A = \{x \in Z_+^3 : x_1 + x_2 + x_3 = 11\}$$

avec $x_1 \leq 3, x_2 \leq 4, x_3 \leq 6$.

Réponse: Soit $A_1 = \{x \in A : x_1 \geq 4\}, A_2 = \{x \in A : x_2 \geq 5\}, A_3 = \{x \in A : x_3 \geq 7\}$.

Nous cherchons la cardinalité de l'ensemble $\bar{A}_1 \cap \bar{A}_2 \cap \bar{A}_3$

$$\begin{aligned} |A \setminus (A_1 \cup A_2 \cap A_3)| &= |A| - |A_1| - |A_2| - |A_3| + |A_1 \cap A_2| + |A_1 \cap A_3| + |A_2 \cap A_3| - |A_1 \cap A_2 \cap A_3| \\ |A| &= {}^{11+3-1}C_{11}. \\ |A_1| &= |\{x \in Z_+^3 : x_1 + x_2 + x_3 = 11, x_1 \geq 4\}| \\ &= |\{x \in Z_+^3 : x_1 + x_2 + x_3 = 7\}| = {}^{7+3-1}C_7. \\ |A_1 \cap A_2| &= |\{x \in Z_+^3 : x_1 + x_2 + x_2 = 11, x_1 \geq 4, x_2 \geq 5\}| \\ &= |\{x \in Z_+^3 : x_1 + x_2 + x_2 = 2\}| = {}^{2+3-1}C_2. \\ A_1 \cup A_2 \cap A_3 &= \emptyset \end{aligned}$$

Donc

$$|A \setminus (A_1 \cup A_2 \cup A_3)| = {}^{13}C_{11} - {}^{7+3-1}C_7 - {}^{6+3-1}C_6 - {}^{4+3-1}C_4 + {}^{2+3-1}C_2 + {}^{1+3-1}C_1 + 0 - 0 = 6.$$

4.5.1 Dénombrement des Fonctions Surjectives

Example 4.15 *Soit N l'ensemble de fonctions $A \to B$ avec $|A| = 6$, $|B| = 3$. Nous voulons trouver le nombre de fonctions surjectives.*

Une fonction n'est pas surjective s'il existe $b \in B$ tel que $b \notin f(A)$.

Soit $B = \{b_1, b_2, b_3\}$.
Soit $Q = \{f \in N : f \text{ est surjective}\}$.
Soit $P_i = \{f : A \to B, b_i \notin f(A)\}$
Soit $Q = N \setminus \cup_{i=1}^{3} P_i$.

$$|N \setminus (P_1 \cup P_2 \cup P_3)| = |N| - \sum |P_i| + \sum |P_i \cap P_j| - |P_1 \cap P_2 \cap P_3|$$

$$|N| = |B|^{|A|} = 3^6!$$

$$|P_i| = |\{f : A \to B \setminus \{b_i\}\}| = (|B| - 1)^{|A|} = 2^6$$

$$|P_i \cap P_j| = |\{f : A \to B \setminus \{b_i, b_j\}\}| = (|B| - 2)^{|A|} = 1^6$$

$$|P_1 \cap P_2 \cap P_3| = 0$$

$$|N \setminus (P_1 \cup P_2 \cup P_3)| = 1 \times 3^6 - 3 \times 2^6 + 3 \times 1^6.$$

Theorem 4.13 *# de fonctions surjectives d'un ensemble de m éléments sur un ensemble de n avec $m \geq n$ est*

$$n^m - {}^{n}C_1(n-1)^m + {}^{n}C_2(n-2)^m + \cdots + (-1)^{n-1} {}^{n}C_{n-1}1^m.$$

4.5.2 Dérangements

Definition 4.1 Un *dérangement* est une permutation qui ne laisse aucun objet dans sa position initiale

Theorem 4.14

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

Preuve: Soit N l'ensemble de toutes les permutations.
P_i = l'ensemble de permutations où $\pi(i) = i$.

Nous cherchons $|N \setminus \cup_{i=1}^{n} P_i|$.

$$|P_{i_1} \cap P_{i_2} \cap \cdots \cap P_{i_k}| = (n-k)!$$

$$\begin{aligned} |N \setminus \cup_{i=1}^{n} P_i| &= n! - {}^nC_1(n-1)! + {}^nC_2(n-2)! \cdots \\ &= n!\left[1 - \frac{1}{1!} + \frac{1}{2!} \cdots + (-1)^n \frac{1}{n!}\right]. \end{aligned}$$

Example 4.16 $n = 3$ $\pi = (312)$ *ou* $\pi = (231)$ $D_3 = 2$

$$3!\left[1 - 1 + \frac{1}{2!} - \frac{1}{3!}\right] = 6\left(\frac{1}{2} - \frac{1}{6}\right) = 2$$

Example 4.17 *Chapeaux! Si cinq personnes distraites partent sans regarder quels chapeaux ils portent, quel est la probabilité qu'aucun n'a pris le bon chapeau?*

$$[1 - 1 + \frac{1}{2!} - \frac{1}{3!} + \frac{1}{4!} - \frac{1}{5!}] = \frac{11}{30}.$$

Chapter 5

Relations de Récurrence

5.1 Etablir une Relation de Récurrence

Nous avons déjà montré que $1+2+\cdots+n = \frac{n(n+1)}{2}$ par induction mathématique. Une autre approche est de définir une fonction

$$f(n) = 1 + 2 + \cdots + n$$

et chercher une formule explicite pour $f(n)$.

Il est clair que

$$\begin{array}{rcl} f(1) & = & 1, \quad \text{et} \\ f(n) & = & f(n-1) + n \quad \text{pour } n > 1. \end{array}$$

On appelle "$f(n) = f(n-1) + n$" une *relation de récurrence*, et "$f(1) = 1$" la *condition initiale.*

En plusieurs cas, la meilleure approche est de

i) Regarder les valeurs de $f(n)$ pour des petites valeurs de n et d'essayer de trouver une relation de récurrence qui satisfait ces valeurs.

ii) Ensuite, il faut trouver un argument qui établit la validité de la récurrence pour tout n.

iii) Finalement, il faut utiliser la relation de récurrence pour établir une expression explicite pour $f(n)$.

Example 5.1 (La tour de Hanoi)
Il y a k anneaux de taille décroissante autour d'un bâton, et deux autres bâtons sans anneaux. Il faut arriver à placer les k anneaux, dans le même ordre, autour d'un autre bâton. Il n'est jamais permis de mettre un anneau au dessus d'un

autre qui est plus petit. Combien de mouvements d'anneaux faut-il au minimum?

Soit S_k = le nombre minimum de mouvements quand il y a k anneaux.

i) Calculer les premières valeurs:

$$S_1 = 1,\ S_2 = 3,\ S_3 = 7,\ \ldots$$

On voit que $S_k = 2S_{k-1} + 1$ pour $k = 2, 3$.

ii) Observer que pour pouvoir bouger le dernier ($k^{ème}$) anneau :

1. *il faut mettre la tour avec $(k-1)$ anneaux sur un des autres bâtons ($\geq S_{k-1}$ mouvements)*
2. *il faut bouger le $k^{ème}$ anneau (1 mouvement);*
3. *il faut remettre la tour avec $(k-1)$ anneaux au-dessus du $k^{ème}$ anneau ($\geq S_{k-1}$ mouvements).*

Il en suit que $S_k \geq 2S_{k-1} + 1$. En plus on peut achever la nouvelle tour en $2S_{k-1} + 1$ mouvements, donc

$$S_k = 2S_{k-1} + 1. \tag{5.1}$$

iii) Ci-dessous on trouvera un moyen systématique pour obtenir une fonction équivalente pour S_k. Ici il est facile de voir que

$$S_k = 2^k - 1$$

satisfait la relation de récurrence et la condition initiale $S_1 = 1$.

Example 5.2 *Quel est le nombre de chaînes binaires (composé de 0 et de 1) qui ne contient pas deux 0 consécutifs.*

Soit C_n l'ensemble de chaînes sans "00" de longueur n, et $f(n) = |C_n|$.

i) Les petites valeurs de n

$n = 1$	0 *ou* 1	$f(1) = 2$
$n = 2$	$01, 10$ *ou* 11	$f(2) = 3$
$n = 3$	$010, 011, 101, 110, 111$	$f(3) = 5$
$n = 4$	$0110, 1010, 1110, 0101, 0111, 1011, 1101, 1111$	$f(4) = 8$

Apparemment $f(n) = f(n-1) + f(n-2)$ pour $n = 3, 4$.

ii) Comment voir que $f(n) = f(n-1) + f(n-2)$ pour tout $n \geq 2$?
Soit A_n les chaînes de longueur n sans deux 0 consécutifs qui terminent en 10, et $p(n) = |A_n|$.
Soit B_n les chaînes de longueur n avec deux 0 consécutifs qui terminent

en 1 et $q(n) = |B_n|$.
Chaque chaîne sans "00" termine soit en 10, soit en 1. Donc $f(n) = p(n) + q(n)$.
En plus on peut ajouter 10 à n'importe quelle chaîne de C_{n-2}. *Donc* $p(n) = f(n-2)$.
De même on peut ajouter 1 à n'importe quelle chaîne de C_{n-1}. *Donc* $q(n) = f(n-1)$.
Finalement on a montré que

$$f(n) = f(n-1) + f(n-2)$$

avec

$$f(1) = 2 \text{ et } f(2) = 3.$$

iii) La suite $f(n) = f(n-1) + f(n-2)$ *avec* $f(1) = f(2) = 1$ *a été étudiée par Fibonacci (voir ci-dessous).*

Example 5.3 (Produit de matrices ou insertion de parenthèses)
Combien y a-t-il de façons de calculer un produit de matrices

$$M = M_1 \times M_2 \times \cdots \times M_k$$

où M_i *est une matrice* $m_i \times m_{i+1}$ *pour* $i = 1 \dots k$ *?*
Soit $T(k) =$ *le nombre de façons différentes quand il y a* k *matrices*

i) Petites valeurs de k

$k = 1$	M_1	$T(1) = 1$
$k = 2$	M_1M_2	$T(2) = 1$
$k = 3$	$M_1(M_2M_3)$, $(M_1M_2)M_3$	$T(3) = 2$
$k = 4$	$M_1(M_2(M_3M_4))$, $M_1((M_2M_3)M_4)$, $(M_1M_2)(M_3M_4)$, $((M_1M_2)(M_3)M_4$, $(M_1(M_2M_3))M_4$	$T(4) = 5$
		etc.

ii) Peut-on trouver une relation de récurrence?
Supposons que la dernière multiplication est le produit

$$(M_1 \dots M_r)(M_{r+1} \dots M_k)$$

indépendamment de la façon de construire $(M_1 \dots M_r)$ *et* $(M_{r+1} \dots M_k)$
Nous avons

$$\begin{aligned} T(k) &= \sum_r T(r)T(k-r) \\ &\equiv T(1)T(k-1) + T(2)T(k-2) + \cdots + T(k-1)T(1) \\ T(4) &= T(1)T(3) + T(2)T(2) + T(3)T(1) \\ &= 5 \end{aligned}$$

iii) Comment trouver $T(k)$?

Example 5.4 *Un algorithme de tri pour $(a_1, a_2, \cdots, a_n)$ qui retourne les mêmes valeurs en ordre croissant. Considérer l'algorithme récursif:*

Tri		$(a_{11} \cdots a_n)$ *If* $n = 1$, *Tri* $= \{a_1\}$, *endif* *Else* $n > 1$
	Tri	$(a_1, \cdots, a_{[n/2]})$
	Tri	$(a_{[n/2]+1}, \cdots, a_n)$
	Merge	$([a_1, \cdots a_{[n/2]}], [a_{[n/2]+1}, \cdots, a_n])$
	End (else)	
	End	

Par exemple, $n = 8$, $A = (2, 6, 3, 1, 7, 5, -2, 4)$.
Tri (2,6,3,1) = (1,2,3,6)
Tri (7,5,-2,4) = (-2,4,5,7)
Merge ([1,2,3,6],[-2,4,5,7]) = (-2,1,2,3,4,5,6,7).

Soit $T(n)$ le nombre de comparaisons, échanges nécessaires pour une liste de longueur n. Si n est pair,

$$T(n) = 2T\left(\frac{n}{2}\right) + n.$$

5.2 Solutions des Relations de Récurrence

Retournons à notre exemple initial avec

$$f(1) = 1 \text{ et } f(n) = f(n-1) + n.$$

Pour trouver une formule explicite, on peut

i) Trouver l'hypothèse que $f(n) = \dfrac{n(n+1)}{2}$ et la vérifier par induction.

ii) Faire l'hypothèse que $f(n) = an^2 + bn + c$ et chercher s'il y a une solution de cette forme

$$f(n) = f(n-1) + n \;\Leftrightarrow\; an^2 + bn + c = a(n-1)^2 + b(n-1) + c + n$$

impliquant

$$\begin{aligned} 0 &= -2an + a - b + n, \ \forall\ n \\ &= n(1-2a) + a - b. \end{aligned}$$

Le coefficient de n doit être 0, donc $(1-2a=0)$, et le terme constant doit être 0, donc $(a-b)=0$. Alors on doit avoir $a=\frac{1}{2}$ et $b=a=\frac{1}{2}$. En plus $f(1)=1$ ssi $a+b+c=1$ qui est satisfait avec $c=0$. Donc

$$f(n)=\frac{1}{2}n^2+\frac{1}{2}n=\frac{1}{2}n(n+1)$$

est une formule explicite.

iii) Une approche plus systématique !

Definition 5.1 Une relation de récurrence *linéaire homogène de degré k à coefficients constants* est une relation de la forme

$$a_n=c_1a_{n-1}+\cdots+c_ka_{n-k}$$

où $c_1\cdots c_k$ sont des réels et $c_k\neq 0$. La relation est de *degré k*.

Example 5.5

i) $P_n=1.11P_{n-1}$ *est une relation de récurrence linéaire à coefficients constants de degré un.*

ii) $F(n)=F(n-1)+F(n-2)$ *est une relation de récurrence linéaire à coefficients constants de degré deux.*

iii) $f(n)=2f(n-1)-1$ *n'est pas homogène*

iv) $g(n)=ng(n-1)$ *n'est pas de coefficients constants*

v) $h_n=h_{n-1}^2+3$ *n'est pas linéaire.*

Definition 5.2 *Equation charactéristique* de $a_n=c_1a_{n-1}+\cdots+c_ka_{n-k}$

$$r^k=c_1r^{k-1}+\cdots+c_kr^0$$

avec racines $\beta_1\cdots\beta_k$.

Theorem 5.1 *Relations de récurrence linéaire d'ordre 1. ($a_n=c_1a_{n-1}$)*
La solution est donnée par la formule

$$a_n=c_1^na_0$$

Theorem 5.2 *Relations de récurrence linéaire d'ordre 2. ($a_n=c_1a_{n-1}+c_2a_{n-2}$)*
Si $r^2-c_1r-c_2=0$ a deux racines β_1,β_2 distinctes, alors

$$a_n=A_1\beta_1^n+A_2\beta_2^n\ \forall\ n=0,1,...$$

Preuve: Montrons d'abord que $A_1\beta_1^n + A_2\beta_2^n$ est une solution pour certaines valeurs de A_1 et A_2.

$$\begin{aligned} a_n &- c_1 a_{n-1} - c_2 a_{n-2} \\ &= A_1\beta_1^n + A_2\beta_2^n - c_1 A_1\beta_1^{n-1} - c_1 A_2\beta_2^{n-1} - c_2 A_1\beta_1^{n-2} - c_2 A_2\beta_2^{n-2} \\ &= A_1(\beta_1^n - c_1\beta_1^{n-1} - c_2\beta_1^{n-2}) + A_2(\beta_2^n - c_1\beta_2^{n-1} - c_2\beta_2^{n-2}) = 0. \end{aligned}$$

Donc $A_1\beta_1^n + A_2\beta_2^n$ est une solution si en plus il satisfait les conditions initiales, c-à-d

$$\begin{aligned} a_0 = a_0^* &= A_1 + A_2 \\ a_1 = a_1^* &= A_1\beta_1 + A_2\beta_2. \end{aligned}$$

Ceci est une système de deux équations en deux inconnues A_1 et A_2. Il y a une solution unique quand $\beta_1 \neq \beta_2$. Pourquoi?

Pour conclure on remarque que la solution est unique une fois les conditions initiales sont satisfaites.

Theorem 5.3 *Relations de récurrence d'ordre 2. ($a_n = c_1 a_{n-1} + c_2 a_{n-2}$) Si $\beta_1 = \beta_2$,*

$$a_n = A_1\beta^n + A_2 n\beta^n$$

Example 5.6 (La série de Fibonacci)
La série $\{F(n)\}_{n=0}^{\infty} \equiv \{0, 1, 1, 2, 3, 5, 8, 13, \ldots\}$ satisfait la relation de récurrence :

$$F(n) = F(n-1) + F(n-2) \text{ pour } n \geq 2$$

avec $F(0) = 0$ et $F(1) = 1$

En essayant $F(n) = \alpha^n$, on obtient :

$$\alpha^n - \alpha^{n-1} - \alpha^{n-2} = 0.$$

En divisant par α^{n-2}, on obtient l'équation charactéristique

$$\alpha^{n-2}\left(\alpha^2 - \alpha - 1\right) = 0.$$

avec racines

$$\alpha = \frac{1 \pm \sqrt{1+4}}{2}.$$

Nous obtenons donc une solution générale :

$$\begin{aligned} F(n) &= c_1\left(\frac{1+\sqrt{5}}{2}\right)^n + c_2\left(\frac{1-\sqrt{5}}{2}\right)^n \\ F(0) &= 0 \Rightarrow c_1 + c_2 = 0 \\ F(1) &= 1 \Rightarrow \frac{c_1 + c_2}{2} + \frac{(c_1 - c_2)\sqrt{5}}{2} = 1 \end{aligned}$$

Alors $c_1 = \frac{1}{\sqrt{5}}$ *et* $c_2 = -\frac{1}{\sqrt{5}}$

Donc

$$F(n) = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n.$$

Si on veut utiliser le Théorème 5.2 directement, on construit l'équation charactéristique

$$r^2 - r - 1 = 0$$

et la suite est la même.

Example 5.7 *Soit* $a_n = 4a_{n-1} - 4a_{n-2}$ *avec* $a_0 = 1$ *et* $a_1 = -1$. *L'équation charactéristique est* $r^2 - 4r + 4 = (r-2)^2 = 0$. *Donc par le Théorème 5.3*

$$\begin{aligned} & a_n = A2^n + Bn2^n \\ & a_0 = A + 0 = 1 \text{ et } a_1 = 2A + 2B = -1 \Rightarrow A = 1 \text{ et } B = -\frac{3}{2} \\ \text{Donc} \quad & a_n = 2^n - \frac{3}{2}n2^n = 2^{n-1}(2-3n) \end{aligned}$$

∎

5.3 Relations de Récurrence avec Fractionnement

Definition 5.3 $f(n) = af(\frac{n}{b}) + g(n)$ est appelé une relation de récurrence *avec fractionnement* (on suppose que $b|n$).

Nous avons vu que pour l'algorithme de tri $f(n) = 2f(\frac{n}{2}) + n$.

Example 5.8
$f(n) = f(\frac{n}{2}) + 2$ *pour un algorithme de fouille binaire.*
$f(n) = 7f(\frac{n}{2}) + \frac{15n^2}{4}$ *pour la multiplication de deux matrices* $n \times n$.

Theorem 5.4 *Soit* f *une fonction croissante qui satisfait*

$$f(n) = af\left(\frac{n}{b}\right) + c$$

lorsque $b|n$, *où* $a \geq 1_q$ *et* $c > 0$. *Alors*

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

Theorem 5.5 *Soit* f *une fonction croissante telle que* $f(n) = af(\frac{n}{b}) + cn^d$ *lorsque* $n = b^k$, *où* $a \geq 1$, $c, d > 0$. *Alors*

$$f(n) = \begin{cases} O(n^d) & si \quad a < b^d \\ O(n^d \log n) & si \quad a = b^d \\ O(n^{\log_n a}) & si \quad a > b^d \end{cases}$$

Example 5.9

a) $f(n) = f(\frac{n}{2}) + 2$. *Par le Théorème 5.4,* $f(n) = O(\log n)$. *En essayant* $f(n) = c\log_2 n$ *et* $n = 2k$, *on obtient*

$$c\log_2 2k = c\log_2 k + 2$$
$$c + c\log_2 k = c\log_2 k + 2.$$

Donc $f(n) = 2\log_2 n$.

b) $f(n) = 2f(\frac{n}{2})+n$. *Par le Théorème 5.5,* $a = 2$, $b = 2$, $c = 1$, $d = 1 \Rightarrow a = b^d$

$$f(n) = O(n\log_2 n).$$

5.4 Fonctions Génératrices

Definition 5.4 La fonction génératrice de la suite de réels $a_0, a_1, \cdots, a_k, \cdots$ est la série

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

Example 5.10 *i) La fonction génératrice de 1,1,1,1,1,1 est*

$$G(x) = 1 + x + \cdots + x^5 = \frac{x^6 - 1}{x - 1}.$$

ii) La fonction génératrice de ${}^nC_k$ *est*

$$G(x) = \sum_{k=0}^{n} {}^nC_k x^k = (1 + x)^n.$$

N.B. Formellement il faut s'assurer que ses équations sont valables pour x *dans un certain domaine. Par exemple* $\frac{1}{1-ax} = 1+ax+\cdots+(ax)^n+\cdots$ *pour* $|x| < \frac{1}{|a|}$ *avec* $a \neq 0$.

Theorem 5.6 *Soit* $f(x) = \sum_{k=0}^{\infty} a_k x^k$ *et* $g(x) = \sum_{k=0}^{\infty} b_k x^k$, *alors*

$$f(x) + g(x) = \sum_{k=0}^{\infty} (a_k + b_k) x^k, \text{ et}$$
$$f(x)g(x) = \sum_{k=0}^{\infty} \left(\sum_{j=0}^{k} a_j b_{k-j} \right) x^k$$

Example 5.11 *Deux dérivations de*

$$g(x) = \sum_{k=0}^{\infty}(k+1)x^k = \frac{1}{(1-x)^2}$$

i) Appliquer le Théorème 5.6 avec $f(x) = q(x) = \frac{1}{1-x}$.

$$f(x)q(x) = \frac{1}{(1-x)^2} = \sum_{k=0}^{\infty}(\sum_{j=1}^{k}1)x^k = \sum_{k=0}^{\infty}(k+1)x^k.$$

ii) $g(x) = \sum_{k=0}^{\infty}(k+1)x^k$.

Alors $G(x) = \int g(x)dx = \sum_{k=0}^{\infty}x^{k+1} = x(\sum_{k=0}^{\infty}x^k) = \frac{x}{1-x}$.

Ensuite

$$\begin{aligned} g(x) = \frac{d}{dx}G(x) &= \frac{d}{dx}\left(\frac{x}{1-x}\right) \\ &= \frac{d}{dx}\left(\frac{1}{1-x} - 1\right) \\ &= \frac{1}{(1-x)^2}. \end{aligned}$$

Example 5.12 *(Fonction de Récurrence)*

$$a_n = 2a_{n-1} + 1, \qquad a_0 = 0.$$

Soit

$$\begin{array}{lcl} G(x) & = & \sum a_n x^n = a_0 + a_1x^1 + a_2x^2 + \cdots \\ -2xG(x) & = & \qquad - 2a_0x^1 - 2a_1x^2 - 2a_2x^3 - \cdots \\ (1-2x)G(x) & = & a_0 + (a_1 - 2a_0)x_1 + (a_2 - 2a)1)x_2 + \cdots \\ & = & \qquad x^1 + x^2 + x^3 + \cdots \\ & = & \frac{x}{1-x} \end{array}$$

Donc

$$\begin{aligned} G(x) &= \frac{x}{(1-x)(1-2x)} = \frac{1}{1-2x} - \frac{1}{1-x} \\ &= \sum_k 2^k x^k - \sum_k x^k = \sum_k (2^k - 1)x^k \end{aligned}$$

et finalement on a

$$a_k = 2^k - 1 \text{ pour } k = 0, 1, \cdots.$$

Vérification pour des valeurs petites:

$$a_0 = 1, a_1 = 1, a_2 = 3, a_3 = 7, a_4 = 15.$$

Example 5.13 *(Multiplication de Matrices)*

$$T(k) = \sum_{j=r}^{k-1} T(r)T(k-r).$$

Soit

$$\begin{aligned} G(x) &= \sum_{k=1}^{\infty} T(k)x^k \\ (G(x))^2 &= (T(1)x + T(2)x^2 + \cdots)(T(1)x + T(2)x^2 + \cdots) \\ &= T(1)T(1)x^2 + (T(1)T(2) + T(2)T(1))x^3 \\ &\qquad +(T(1)T(3) + T(2)T(2) + T(3)T(1))x^4 + \cdots \end{aligned}$$

Alors $(G(x))^2 = \sum_{k=2}^{\infty}(T(1)T(k-1) + \ \cdots \ + T(k-1)T(1))x^k$
Mais $T(1)T(k-1) + \cdots + T(k-1)T(1) = T(k)$ *par la relation de récurrence, donc* $(G(x))^2 = \sum_{k=2}^{\infty} T(k)x^k$.
Donc $(G(x))^2 = G(x) - x$
Une résolution formelle de ceci comme équation quadratique donne :

$$G(x) \ = \ \frac{1 \pm \sqrt{1-4x}}{2}$$

Parce que $G(0) = 0$, *nous prenons* $G(x) = \frac{1-\sqrt{1-4x}}{2}$ *et en faisant le développement de* $(1-4x)^{\frac{1}{2}}$ (Théorème du binôme)[1], *nous obtenons*

$$\begin{aligned} G(x) &= \frac{1}{2} - \frac{1}{2}(1-4x)^{1/2} \\ &= \frac{1}{2} - \frac{1}{2}\left[1 + (-4x)(-\frac{1}{2}) + (-4x)^2(-\frac{1}{2})(-\frac{3}{2})/2! + \cdots\right] \\ &= -\sum_{n=1}^{\infty} x^n \frac{1}{2}\left[(-4)^n \frac{\frac{1}{2}(-\frac{1}{2})\cdots(\frac{1}{2}-(n-1))}{n!}\right] \\ &= \sum_{n=1}^{\infty} x^n \left[\frac{(2n-3)(2n-5)\cdots 1 \times 2^{n-1}}{n!}\right] \\ &= \sum_{n=1}^{\infty} x^n \left[\frac{(2n-2)!}{(n-1)!\ n!}\right] \\ T(n) &= \frac{(2n-2)!}{(n-1)!\ n!} \end{aligned}$$

[1] $(1+ax)^b = 1 + \cdots \sum_{n=1}^{\infty} \frac{a^n b(b-1)\cdots(b-n+1)}{n!} x^n$.

Vérification : $T(4) = \frac{6!}{3!\,4!} = 5$ (Les nombres de Catalan)
Observons que $T(k)$ est le nombre de façons différentes d'insérer des parenthèses, ce qui revient dans la construction d'expressions dans un langage.

Une autre possibilité est de prendre

$$G(x) \equiv \sum_{k=0}^{\infty} F(x)x^k$$

et de calculer $(1 - x - x^2)G(x)$.
Une autre question qu'il est naturel de poser concerne la vitesse de croissance de $F(n)$ quand n devient grand.

Chapter 6

Algorithmes et Complexité

Ici nous étudions quelques problèmes types pour mieux les analyser.

6.1 Algorithme de Fouille Binaire

$(a_1, \cdots, a_n)$ est une liste de nombres en ordre non décroissant, ou une liste de mots en ordre alphabétique. Vous voulez vérifier si x se trouve dans la liste ou pas.
L'idée de l'algorithme est de regarder au milieu de la liste, l'élément $a_{\lfloor \frac{n+1}{2} \rfloor}$.
Si $x = a_{\lfloor \frac{n+1}{2} \rfloor}$, où $n = 1$ et $x \neq a$, stop.
Si $x > a_{\lfloor \frac{n+1}{2} \rfloor}$, regarder seulement la partie de liste $a_{\lfloor \frac{n+1}{2} \rfloor + 1}, \cdots, a_n$.
Si $x < a_{\lfloor \frac{n+1}{2} \rfloor}$, regarder seulement la partie de liste $a_1, \cdots a_{\lfloor \frac{n+1}{2} \rfloor - 1}$.

Soit $T(n)$ le nombre d'opérations (comparaisons) nécessaires dans le pire des cas pour vérifier une liste de taille n.
Trouver une récurrence pour $T(n)$. (On suppose que l'élément $a_{\lfloor \frac{n+1}{2} \rfloor}$ au milieu de la liste peut se trouver dans un temps constant, c'est-à-dire 1 opération).

Example 6.1 *1 3 4 5 6 13 27 43 56 82 n=10*

	$x = 42$						
$a_5 = 6$	$x > a_5$	*Liste*$(n = 5)$	*13*	*27*	*43*	*56*	*82*
$a'_3 = 43$	$x < 43$	*Liste* $(n = 2)$	*13*	*27*			
$a''_1 = 13$	$x > 13$	*Liste*$(n = 1)$	*27*				
$a'''_1 = 27$	$x \neq 27$		*Stop*				

6.2 Le plus grand diviseur de deux entiers

Ici nous étudions l'algorithme Euclidien pour trouver le plus grand diviseur de deux entiers : c-à-d trouver

$$\text{pgcd}(a,b) = \max\{r : r|a,\ r|b\} \text{ où } a,b \in Z_+^1,$$

(Si $a = 0,\ \text{pgcd}(a,b) = b$)

Considérons le problème d'un point de vue mathématique

Proposition 6.1 *Soit $a,b \in Z_+^1$ avec $a \geq b$ et $c = a - \lfloor \frac{a}{b} \rfloor b$.*

1. *Si $c \neq 0,\ \text{pgcd}(a,b) = \text{pgcd}(b,c)$.*
2. *Si $c = 0,\ \text{pgcd}(a,b) = b$.*

Preuve.

1. Soit $r = \text{pgcd}(a,b)$, $s = \text{pgcd}(b,c)$, et $d = \lfloor \frac{a}{b} \rfloor \in Z_+^1$.
   Parce que $c = a - db$ et r divise a et b, r doit diviser c.
   Donc r est un diviseur de b et de c, et par définition de s, $r \leq s$.

   En utilisant $a = c + db$, l'argument identique nous donne que $s|a$ et donc $s \leq r$.
2. Si $c = 0$, $a = db$, et $\text{pgcd}(a,b) = b$. ∎

Cette proposition donne une indication sur l'algorithme. Parce que $c < b \leq a$, nous allons appliquer la Proposition 1 d'abord à (a,b), ensuite à (b,c) etc.

6.2.1 Algorithme d'Euclide 1

Initialisation
Données : $a,b \in Z_+^1$.
Ordonner a et b pour que $a \geq b \geq 0$. Si $b = 0$, $\text{pgcd}(a,b) = a$.
$t = 1$. $\quad a_1 = a, \quad b_1 = b$
Itération t

$$\begin{aligned} d_t &= \lfloor \frac{a_t}{b_t} \rfloor \\ c_t &= a_t - d_t b_t \end{aligned}$$

Si $c_t = 0$, Stop.
Mettre $T = t$. $\text{pgcd}(a,b) = b_T$. Sortie (T, b_T)
Sinon $a_{t+1} = b_t,\ b_{t+1} = c_t$.
Augmenter t.

Theorem 6.2 *L'algorithme est correct.*

Preuve par induction.

Hypothèse Inductive:

$\text{pgcd}(a_t, b_t) = \text{pgcd}(a, b)$,
pour $t = 1,\ a_t = a,\ b_t = b$ et l'hypothèse est satisfaite.

Pas inductif :

Supposons $\text{pgcd}(a_t, b_t) = \text{pgcd}(a, b)$

Par la proposition $\text{pgcd}(a_t, b_t) = \text{pgcd}(b_t, c_t) = \text{pgcd}(a_{t+1},\ b_{t+1})$.

Donc par induction,

$$\text{pgcd}(a_T, b_T) = \text{pgcd}(a, b),$$

Parce que $c_T = 0$, il découle de la Proposition 1 que $\text{pgcd}(a_T, b_T) = b_T$. ∎

Example 6.2

$$(a, b) = (51, 36)$$

t	a_t	b_t	d_t	c_t
1	*51*	*36*	*1*	*15*
2	*36*	*15*	*2*	*6*
3	*15*	*6*	*2*	*3*
4	*6*	*3*	*2*	*0*

$$T = 4,\ \text{pgcd}(51, 36) = 3$$

Au moins deux questions importantes se posent.

1. **Une bonne preuve**
   Peut-on vérifier que 3 est la solution correcte sans refaire les calculs?
   On peut vérifier que $3|51$ et $3|36$. Donc que $3 \leq \text{pgcd}(51, 36)$,
   mais comment prouver que $3 = \text{pgcd}(51, 36)$?
2. **Efficacité**. Combien d'itérations faut-il?
   D'abord on observe que le nombre d'itérations $\leq b$, parce que $b_1 > b_2 > \ldots > 1$. Donc l'algorithme est fini.

Nous essayons de répondre la première question.

6.2.2 Une Preuve Rapide

Proposition 6.3 *S'il existe* $p, q \in Z^1 \backslash \{0\},\ r \in Z^1_+ \backslash \{0\}$ *tel que* $r = pa + qb$, *alors* $pgcd(a, b)|r$.

Preuve.
Soit $s = \text{pgcd}(a, b),\ \ a = k_1 s,\ \ b = k_2 s$ avec $k_1, k_2 \in Z^1$.
Donc $r = (pk_1 + qk_2)s$ avec $pk_1 + qk_2 \in Z^1$ et $s|r$. ∎

Cette proposition indique la manière dont on peut donner une bonne preuve que le résultat de l'algorithme est correct.
Il faut donner les trois entiers p, q et r Alors il suffira de vérifier:

1. $r = pa + qb$ ce qui implique $\text{pgcd}(a,b)|r$.
2. $r|a,\ \ r|b$ ce qui implique $r|\,\text{pgcd}(a,b)$.

et alors nous sommes sûrs que $r = \text{pgcd}(a,b)$.

Question. Est-ce que l'Algorithme 1 nous permet de trouver p et q tel que $pa + qb = r$.

Réponse.

$$\begin{aligned} r &= c_3 = a_3 - 2b_3 \\ &= b_2 - 2c_2 \\ &= b_2 - 2(a_2 - 2b_2) \\ &= -2a_2 + 5b_2 \\ &= -2b_1 + 5c_1 \\ &= -2b_1 + 5(a_1 - b_1) \\ &= 5a_1 - 7b_1 \\ &= 5a - 7b \end{aligned}$$

Vérification. $5 \times 51 - 7 \times 36 = 3$.

Maintenant nous pouvons modifier l'algorithme pour qu'à chaque moment on ait p', q' tel que $c_t = p'a + q'b$.

6.2.3 Algorithme 2

Initialisation

Données $(a,b) \in Z_+^2$ avec $a \geq b > 0$.
$(c_{-1}, p_{-1}, q_{-1}) = (a, 1, 0), \quad (c_0, p_0, q_0) = (b, 0, 1)$
$a_1 = a,\ \ b_1 = b$
$t = 1$

Itération t

$$\begin{aligned} d_t &= \lfloor \frac{a_t}{b_t} \rfloor \\ c_t &= a_t - d_t b_t \\ p_t &= p_{t-2} + d_t p_{t-1} \\ q_t &= q_{t-2} + d_t q_{t-1} \end{aligned}$$

Si $c_t = 0$, stop .
  Mettre $T = t$.
  $\text{pgcd}(a,b) = b_T$
  $(p,q) = (-1)^T (p_{T-1}, q_{T-1})$
Sinon $a_{t+1} = b_t,\ b_{t+1} = c_t$
Augmenter t;

Proposition 6.4 $c_t = (-1)^{t+1}[p_t a - q_t b]$
pour $t = -1, \ldots, T$.

Preuve.
Par induction

$$\begin{array}{lllll} t=-1 & c_{-1} & = & [1\times a-0\times b] & =a \\ t=0 & c_0 & = & -[0\times a-1\times b] & =b \end{array}$$

Si l'hypothèse est satisfaite pour $t<k$,

$$\begin{array}{rcll} c_k & = & a_k-d_kb_k & \\ & = & c_{k-2}-d_kc_{k-1} & \\ & = & (-1)^{k-1}(p_{k-2}a-q_{k-2}b) & -(-1)^k d_k(p_{k-1}a-q_{k-1}b) \\ & = & (-1)^{k-1}[(p_{k-2}+d_kp_{k-1})a & -(q_{k-2}+d_kp_{k-2})b] \\ & = & (-1)^{k-1}[\ \ p_ka & \ \ -q_kb] \end{array}$$

et le résultat est vrai pour $t=k$. ∎

Maintenant la bonne preuve est disponible parce que $\text{pgcd}(a,b)=b_T = c_{T-1} = (-1)^T(p_{T-1}a-q_{T-1}b)$, et il suffit de prendre $p=(-1)^Tp_{T-1}$ et $q=(-1)^{T-1}q_{T-1}$.

Example 6.3

t	a_t	b_t	d_t	c_t	p_t	q_t
-1				*51*	*1*	*0*
0				*36*	*0*	*1*
1	*51*	*36*	*1*	*15*	*1*	*1*
2	*36*	*15*	*2*	*6*	*2*	*3*
3	*15*	*6*	*2*	*3*	*5*	*7*
4	*6*	*3*	*2*	*0*	*12*	*17*

Vérification :
$(p,q)=(-1)^4(p_3,-q_3)=(5,-7)$ *et* $5\times 51-7\times 36=3$. *Observons aussi que*

$$\frac{0}{1}\le\frac{2}{3}\le\frac{12}{17}\le\frac{36}{51}\le\frac{1}{1}\le\frac{1}{0}.$$

Tournons maintenant à la deuxième question.

6.2.4 Efficacité de l'Algorithme

1. Efficacité
2. Bonne preuve

Combien d'itérations faut-il?
Aussi pour être efficace, et aussi pour que notre "bonne preuve" soit vraiment bonne, il faut être sûr que les chiffres p_t, q_t ne sont pas énormes!

Proposition 6.5

1. $b_{t-1}\ge b_t+b_{t+1}$
2. $b_{T+1-t}\ge F(t)$ *(où* $F(t)$ *est le* t^e *nombre de Fibonacci)* $t=1,\dots T$

3. $T = O(\log b)$

4. $(p_t q_{t+1} - p_{t+1} q_t) = (-1)^{t+1}$ *pour* $t = -1, 0, \ldots\ T$

5. $\text{pgcd}(p_t, q_t) = 1$ *pour* $t = -1, 0, \ldots, T$

6. $q_T = \dfrac{a}{\text{pgcd}(a,b)}$, $p_T = \dfrac{b}{\text{pgcd}(a,b)}$

Preuve.

1. $\begin{array}{rcl} a_t & = & d_t b_t + c_t \text{ est équivalent à} \\ b_{t-1} & = & d_t b_t + b_{t+1} \\ & \geq & b_t + b_{t+1} \text{ parce que } d_t \geq 1 \end{array}$

2. Par induction
$\begin{array}{rcl} b_T & = & \text{pgcd}(a,b) \geq 2 = F(2) \\ b_{T-1} & \geq & \text{pgcd}(a,b) \geq 1 = F(3) \\ b_{T+1-t} & \geq & b_{T+2-(t-1)} + b_{T+2-(t-2)} \text{ par 1.} \\ & \geq & F(t) + F(t-1) \text{ par l'hypothèse inductive} \\ & = & F(t+1) \end{array}$

3. Parce que $b = b_1 \geq F(T)$ et $F(T) \geq \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2} \right)^T - 1$,
$\frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2} \right)^T - 1 \leq F(T) + 1 \leq b + 1 \leq 2b - 1$ pour $b \geq 2$.
Soit $\alpha \in R^1$ tel que $\frac{1+\sqrt{5}}{2} = 2\alpha$.
Alors $\frac{1}{\sqrt{5}} (2\alpha)^T \leq 2b$
$T(1 + \log_2 \alpha) \leq 1 + \log_2 b + \frac{1}{2} \log_2 5$,
$T =$O$(\log b)$ ∎

Parce que $q_1 \leq q_2 \ \ldots \ \leq q_T$, par le point (6.) de la proposition 5, on voit que les valeurs de p_t et q_t restent $\leq a$.

Theorem 6.6 *L'algorithme Euclidien se termine en* $O(\log(b))$ *itérations et les chiffres dans les calculs intermédiaires ne depassent jamais* $\max(a, b)$

Example 6.4 $a = 51$, $b = 36$, $T = 4$
$(b_1, b_2, b_3, b_4) = 36, 15, 6, 3$

1. *La Proposition 6.5 confirme que* $15 \geq 6 + 3$, $36 > 15 + 6$, *etc.*

2. *Elle confirme également que* $b_1 \geq F(T)$

Example 6.5 $a = Fib(10) = 55, b = Fib(9) = 34$
$b_1 = 34, b_2 = 21, b_3 = 13, b_4 = 8, b_5 = 5, b_6 = 3, b_7 = 2, b_8 = 1, T = 8$

6.3 Remarque sur la complexité

Nous jugeons qu'un tel algorithme est *efficace* parce que le nombre de bits pour introduire les données est :

$$L = \lceil \log_2 a \rceil \ + \ \lceil \log_2 b \rceil$$

et le nombre d'itérations est $O(L)$ et la taille des chiffres et l'espace utilisé est $O(L)$. [1]

Un algorithme est *polynomial* s'il existe un k tel que le temps de calcul est $O(L^k)$, c.à.d le temps est une fonction polynomiale en L. Dans la suite nous verrons des problèmes pour lesquels nous connaissons des *algorithmes exponentiels*

$$\text{Temps} \ = \ O(2^L)$$

mais aucun algorithme polynomial n'est connu.

6.4 Programmes Récursives et Itératives

6.4.1 Algorithme d'Euclide

Considérons brièvement la manière de programmer les deux algorithmes.

Nous l'avons décrit de façon itérative. A chaque moment il suffit de disposer des données (a, b) et des valeurs $(a_t, b_t, p_t, q_t, p_{t-1}, q_{t-1})$ pour pouvoir arrêter l'algorithme et le relancer plus tard.

Algorithme 3

Une description très compacte de l'algorithme est donnée. La définition est *récursive* mais l'algorithme est de nouveau itératif.

Define pgcd(a, b)
  If $b = 0$ *then* a
  Else pgcd $\big(b, a - \lfloor \frac{a}{b} \rfloor b\big)$.

6.4.2 Calcul du factorielle

Factorielle $f(n) = n!, \quad n \in Z^+ \backslash \{0\}$

Considérons ici deux programmes différentes pour calculer $n!$

Programme 1 Define $f(n)$
  If $n = 1$, then $f(n) = 1$
  Else $f(n) = nf(n-1)$

Voici comment évolueront les calculs :

$$\begin{aligned} 6! &= 6 \times (5!) \\ &= 6 \times (5 \times (4!)) \end{aligned}$$

[1] $\lceil x \rceil$ = plus petit entier $\geq x$.

$$\begin{aligned} &= 6 \times (5 \times (4 \times (3!))) \\ &= 6 \times (5 \times (4 \times (3 \times (2!)))) \\ &= 6 \times (5 \times (4 \times (3 \times (2 \times 1)))) \\ &= 6 \times (5 \times (4 \times (3 \times (2))) \\ &= (6 \times (5 \times (4 \times 6))) \\ &= 6 \times (5 \times 24) \\ &= 6 \times 120 \\ &= 720 \end{aligned}$$

Ceci est une procédure récursive linéaire.

Programme 2 t = itération
p=produit
Define $f(n) = n! = g(1, 1, n)$
Define $g(p, t, n)$
If $t > n$, then p
Else $g(t \times p, t + 1, n)$

Maintenant on obtient :

$$\begin{aligned} f(6) &= g(1, 1, 6) \\ &= g(1, 2, 6) \\ &= g(2, 3, 6) \\ &= g(6, 4, 6) \\ &= g(24, 5, 6) \\ &= g(120, 6, 6) \\ &= g(720, 7, 6) \\ &= 720 \end{aligned}$$

Ceci est une procedure itérative linéaire.

6.4.3 Nombres de Fibonnaci

Programme 1
Define $F(n)$
If $n = 0$,then 0
If $n = 1$, then 1
Else $F(n - 1) + F(n - 2)$

Calculons $F(5)$ par substitution :

Combien de calculs faut-il faire? Voir la Figure 6.4.3 qui représente les calculs de l'algorithme. Il y a un calcul pour chaque noeud de l'arbre. Mais il y a $F(n)$ noeuds terminaux de valeur 1 dans l'arbre. Donc $F(n)$ calculs. Mais

$$F(n) \geq \frac{1}{\sqrt{5}} \left(\frac{1 + \sqrt{5}}{2} \right)^n - 1$$

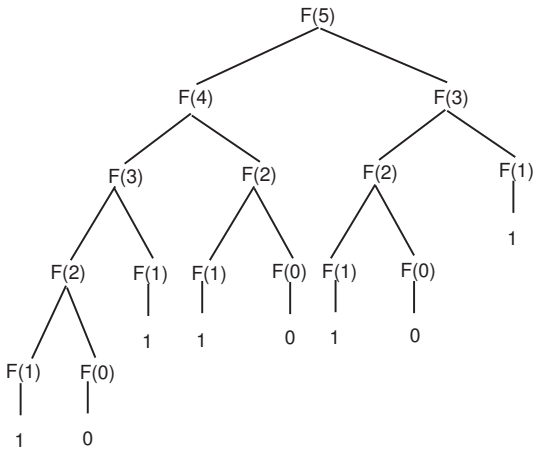


Figure 6.1: Il faut construire cet arbre, et après remonter pour faire les additions (ceci est équivalent à faire la somme des valeurs sur les feuilles). On obtient $F(5) = 5$, mais nous avons calculé $F(3)$ deux fois, $F(2)$ trois fois, etc.

Donc cet algorithme est exponentiel en n.

Programme 2 Define $Fib(n) = Fib\ It(1, 0, n; n)$

Define $Fib\ It(a, b, count; n)$

If $count = 1$, then a

Else $Fib\ It(a + b, a, count - 1; n)$

Pour calculer $Fib(5)$ par substitution

$$\begin{aligned} Fib(5) &= FibIt(1, 0, 5; 5) \\ &= FibIt(1, 1, 4; 5) \\ &= FibIt(2, 1, 3; 5) \\ &= FibIt(3, 2, 2; 5) \\ &= FibIt(5, 3, 1; 5) \\ &= 5 \end{aligned}$$

Ici l'algorithme est linéaire itératif (et donc polynomial) en n.

Observons que donner une preuve n'est pas évident pour ces deux derniers exemples. Nous avons plutôt voulu examiner comment différentes définitions récursives de fonctions mènent à des algorithmes fort différents. Donc c'est important de savoir comment un langage de programmation particulier traite des fonctions définies par récursion.

6.5 Introduction à la Programmation Dynamique

La programmation dynamique est une approche particulière pour certains problèmes d'optimisation. Ces problèmes sont caractérisés par la propriété suivante.

Etant donné une solution optimale du problème, une partie de cette solution est nécessairement la solution optimale d'un sous-problème du problème initial.

Par exemple

i) Si le chemin le plus court de A B passe par C, alors la partie de ce chemin entre A et C est le plus court chemin de A à C.

ii) Si, étant donné une séquence $a_1, a_2, \cdots, a_n$, si $a_{j_1}, a_{j_2}, \cdots, a_{j_r}$ est une sous-séquence optimale de $a_1, a_2, \cdots, a_{j_r}$, alors $a_{j1}, \cdots, a_{j_{r-1}}$ est une sous-séquence optimale de $a_1, a_2, \cdots, a_{j_{r-1}}$.

Cette propriété est appelée le *Principe d'Optimalité.*

Nous allons voir quatre exemples. Le premier, qui est une variante du problème du plus court chemin, va nous permettre d'introduire la terminologie qui est souvent utilisée en programmation dynamique.

Le problème du "Stagecoach"

Il faut aller du noeud A (état 1) à l'étape 0 jusqu'au noeud B (état 1) à l'étape 4. Aux étapes intermédiaires on peut passer par différents noeuds (états), voir Figure 1. Les valeurs $\{c_{ij}^k\}$ représentent le temps de passage de l'état i à l'étape $k-1$ à l'état j à l'étape k.

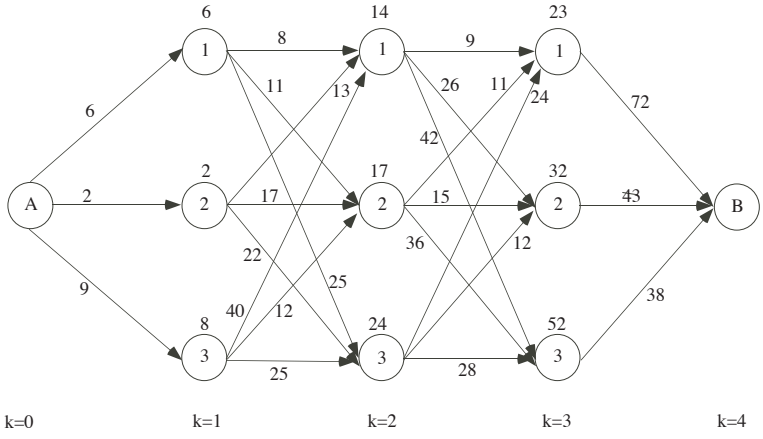


Figure 1

Observons que si $A213B$ est un plus court chemin de A à B , $A213$ est un court chemin de A à l'état 3 de l'étape 3.

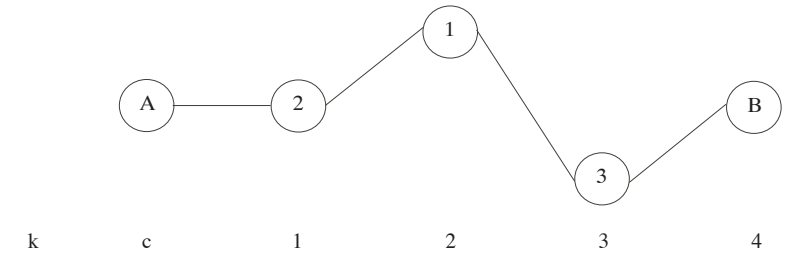


Figure 2

Récurrence

Soit $f_k(j) \equiv$ la longueur du plus court chemin de A à l'état j de l'étape k. Nous avons

$$\begin{array}{rcl} f_0(A) & = & 0 \\ f_k(j) & = & \min\limits_{i \text{ à l'étape } k-1} \{f_{k-1}(i) + c_{ij}^k\} \\ p_k(j) & = & \text{argmin } \{f_{k-1}(i) + c_{ij}^k\} \end{array}$$

Autrement dit, $f_k(j) = f_{k-1}(p_k(j)) + c^k_{p_k(j),j}$.

Exemple numérique

Nous observons que

$$f_1(1) = 6, f_1(2) = 3, f_1(3) = 9.$$

Ensuite

$$\begin{array}{rcl} f_2(1) & = & \min[f_1(1) + c_{11}^2, f_1(2) + c_{21}^2, f_1(3) + c_{31}^2] \\ & = & \min[6 + 8, 2 + 13, 9 + 40] = 14 \\ p_2(1) & = & 1 \end{array}$$

f	k	0	1	2	3	4
j	1	0	6	14	23	75
	2	∞	2	17	32	
	3	∞	9	24	52	

p	0	1	2	3	4
1	–	1	1	1	2
2	–	1	1	2	–
3	–	1	2	3	–

- Valeur optimale
  Ceci nous permet de calculer la longueur du plus court chemin de A à B, c'est-à-dire,

$$f_4(1) = 75$$

- Solution optimale
  Pour trouver la solution optimale, il faut faire le retour à partir de B

$$\begin{array}{ll} p_4(1) = 2 & \text{indique que le dernier arc est 2,B} \\ p_3(2) = 2 & \text{indique que le dernier arc est 2,2} \\ p_2(2) = 1 & \text{indique que le dernier arc est 1,2} \\ p_1(1) = 1 & \text{indique que le dernier arc est A,1} \end{array}$$

  Donc le chemin optimal est $A122B$ de longueur 75.

Le Problème de Sous-Séquence Strictement Croissant Maximum

Etant donné une séquence $a_1, a_2, \cdots, a_n$ de nombres,

$$a_{j_1}, a_{j_2}, \cdots, a_{j_r}$$

est une sous-séquence strictement croissante si

$$1 \leq j_1 < j_2 < \cdots j_r \leq r$$

et

$$a_{j_1} < a_{j_2} < \cdots < a_{j_r}.$$

Principe d'optimalité

$$a_j : \quad 9 \quad 2 \quad 8 \quad 7 \quad 3 \quad 5 \quad 4 \quad 1 \quad 3$$

Etant donné que $2\,3\,4 = a_2 a_5 a_7$ est une SSC maximum terminant en avec 4 à la position 7 ($a_7 = 4$), 23 est une SSC max terminant en 3 à la position 5.

Récurrence

Soit $f(i)$ la longueur d'une SSC de longueur maximum terminant avec a_i et $g(i)$ la longueur d'une SSC maximum de $a_1, \cdots, a_i$. Evidemment $g(i) = \max_{j:j\leq i} f(j)$.

$$\begin{array}{rcl} f(0) & = & 0 \\ f(j) & = & \max\limits_{a_i < a_j, i<j} [1 + f(i)] \\ p(j) & = & {\scriptstyle a_i < a_j, i<j}[1 + f(i)] \\ g(n) & = & \max_j f(j). \end{array}$$

Valeur optimale $g(n)$

Exemple numérique Par exemple, avec $f(1) = f(2) = 1$, $f(3) = f(4) = f(5) = 2$,

$$\begin{array}{rcl} f(6) & = & \min\{1 + f(i) : i < 6, a_i < 5\} \\ & = & \min[1 + f(2), 1 + f(5)] = 3 \\ p(6) & = & 5 \end{array}$$

On obtient

i	=	0	1	2	3	4	5	6	7	8	9
$f(i)$	=	0	1	1	2	2	2	3	3	1	2
$p(i)$	=	0	0	0	2	2	2	5	5	0	2

Avec $g(9) = 3 = f(6)$, on obtient $p(6) = 5$, $p(5) = 2$, $p(2) = 0$, donc une SSC maximum $a_2, a_5, a_6 = 2\,3\,5$.

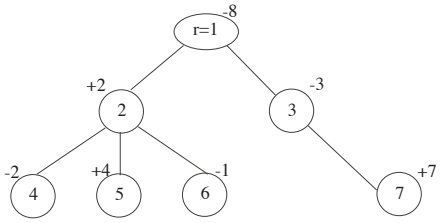
Sous-arboresence d'une Arboresence de Valeur Maximum

Etant donné une arborescence orientée $T = (V, A)$ avec racine $r = 1$, et des valeurs c_j pour $j \in V$, il faut trouver une sous-arborescence T' de racine r telle que

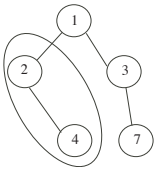
$$\sum_{v \in V(T')} c_v$$

est maximisé.

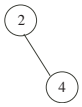
Principe d'Optimlité



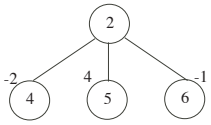
Si



est une sous-arborescence optimale de T, alors



est une sous-arborescence optimale du sous-arbe de racine 2



Récurrence

Soit $H(v)$ la valeur optimale d'une sous-arborescence $U(v)$ de racine v du sous-arbe $T(v)$. Il y a deux cas.
Soit

$$v \notin U(v) \text{ et } H(v) = 0$$

soit

$$v \in U(v) \text{ et } H(v) = c_v + \sum_{w \in S(v)} H(w),$$

où $S(v)$ est l'ensemble des fils (successeurs directes) de v. Donc

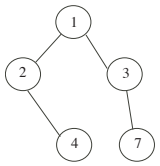
$$H(v) = \max[0, c(v) + \sum_{w \in S(v)} H(w)].$$

v	7	6	5	4	3	2	1
$H(v)$	7	0	4	0	4	6	2

La valeur optimale $z = H(1) = 2$.
Pour trouver le sous-arbre optimal:

$$\begin{array}{lll} & H(1) = 2 & \text{Donc } 1 \in U(1) \\ S(1) = \{2, 3\} & & \\ & H(2) = 2 & \text{Donc } 2 \in U(1) \\ S(2) = \{4, 5, 6\} & & \\ & H(3) = 4 & \text{Donc } 3 \in U(1) \\ S(3) = \{7\} & & \\ & H(4) = 0 & \text{Donc } 4 \notin U(1) \\ & H(5) = 4 & \text{Donc } 5 \in U(1) \\ & H(6) = -1 & \text{Donc } 6 \notin U(1) \\ & H(7) = 7 & \text{Donc } 7 \in U(1) \end{array}$$

Donc $U(1) = \{1, 2, 3, 4, 7\}$



6.5.1 Multiplication de Matrices

("Diviser pour conquérir")

6.5.2 Problème

Données: k matrices $M_1, M_2, \ldots M_k$ avec M_i une matrice $m_i \times m_{i+1}$.

(Nous supposons que pour multiplier deux matrices A $m \times n$ et B $n \times p$ il faut mnp calculs.)

En quel ordre faut-il multiplier les matrices pour minimiser le nombre de calculs?
Rappelons qu'il y a $T(n) = \frac{(2n-2)!}{n!(n-1)!}$ façons différentes de multiplier n matrices. Parce que $T(n) = O(2^n)$, l'énumeration de toutes les possibilités est une *mauvaise* approche.

Example 6.6 *Soit* $M_1 = 10 \times 20, M_2 = 20 \times 5, M_3 = 5 \times 30$.

1. $(M_1, M_2)M_3$*: Nombre de multiplicateurs:* $20 \times 20 \times 5 + 10 \times 5 \times 30 = 2500$
2. $M_1(M_2M_3)$*: Nombre de multiplicateurs:* $20 \times 5 \times 30 + 10 \times 20 \times 30 = 9000$.

Donc il est clair que $(M_1, M_2)M_3$ *est préférable.* ∎

6.5.3 Observation

Si nous avons trouvé la façon optimale de multiplier les matrices
$M_i, M_{i+1}, \ldots, M_j$
et que le dernier produit est de la forme
$(M_iM_{i+1} \ldots M_r)(M_{r+1} \ldots M_j)$ avec $i < r < j$,
les façons de multiplier $M_i, \ldots M_r$ et de multiplier $M_{r+1}, \ldots M_j$ doivent aussi être optimale.

Soit $G(1, k)$ le nombre minimum de calculs nécessaires pour multiplier $M_1, \ldots M_k$
Par l'observation : pour $i < j$

$$\begin{aligned} G(i,i) &= 0 \\ G(i,j) &= \min_{i \le r < j} [G(i,r) + m_i m_{r+1} m_{j+1} + G(r+1,j)] \end{aligned}$$

Si on utilise cette définition dans une procédure récursive, c'est désastreux. D'autre part, il est simple d'organiser les calculs de façon itérative.

Pour $\lambda = 1, \ldots, k-1$
Pour $i = 1, \ldots, k-\lambda$
Calculer $G(i, j)$ avec $j = i + \lambda$
Solution : $G(1, k)$.

Example 6.7 $k = 4$; $(m_1, m_2, m_3, m_4, m_5) = (2, 6, 4, 1, 3)$
$\lambda = 1$

$$\begin{aligned} G(1,2) &= \{G(1,1) + 2.6.4 + G(2,2)\} &= 48, \\ G(2,3) & &= 24, \\ G(3,4) & &= 12 \end{aligned}$$

$\lambda = 2$

$$\begin{aligned} G(1,3) &= \min\{G(1,1) + m_1 m_2 m_4 + G(2,3), G(1,2) + m_1 m_3 m_4 + G(3,3)\} \\ &= \min\{0 + 12 + 24, 48 + 8 + 0\} = 36 \ (r = 1) \\ G(2,4) &= \min\{0 + 72 + 12, 24 + 18 + 0\} = 42 \ (r = 2) \end{aligned}$$

$\lambda = 3$

$$\begin{aligned} G(1,4) &= \min\{G(1,1) + m_1 m_2 m_5 + G(2,4), G(1,2) + m_1 m_3 m_5 \\ &\quad + G(3,4), G(1,3) + m_1 m_4 m_5 + G(4,4)\} \\ &= \min\{(0 + 36 + 42), (48 + 24 + 12), (36 + 6 + 0)\} = 42 \ (r = 3) \end{aligned}$$

Donc la solution optimale demande 42 multiplications.

Quelle est la solution?

Parce que $r = 3$ *dans le calcul de* $G(1,4)$*, il faut multiplier* $(M_1 M_2 M_3)$ *avec* M_4*. (6 calculs)*

Parce que $r = 1$ *dans le calcul de* $G(1,3)$*, il faut multiplier* M_1 *avec* $(M_2 M_3)$*. (12 calculs)*
Pour multiplier $M_2 M_3$ *il faut 24 calculs.*
Solution $(M_1(M_2 M_3))M_4$ *avec 42 calculs.* ∎

Chapter 7

Connexité et Arbres

Dans cette partie sur les graphes, nous abordons seulement un petit nombre de questions. Ici, après quelques définitions de base des graphes non orientés, nous allons étudier la *connexité* d'un graphe. Presque tout problème sur un graphe peut être décomposé en de plus petits problèmes sur les *composantes connexes* du graphe, donc la première question est de trouver cette décomposition. Les plus petits graphes connexes sont les *arbres*. Un arbre établit un chemin entre chaque paire de nœuds d'un graphe, et sert naturellement comme modèle pour un réseau minimal de communication entre ordinateurs, "switching centers" en télécommunication, un poste MT/BT et les utilisateurs domestiques, etc. Etant donné les coûts d'installation des lignes entre chaque paire de nœuds, un problème naturel est de chercher *l'arbre de coût minimum*. Nous étudions deux algorithmes pour ce problème.

Un problème qui revient chaque fois qu'on étudie un objet abstrait est de reconnaître si deux objets sont similaires ou identiques. Le chapitre 8 s'ouvre en formalisant cette idée avec le concept d'*isomorphisme* entre deux graphes. Nous discutons aussi différentes *représentations* d'un graphe. Ensuite, nous donnons quelques définitions des graphes orientés ou *digraphes* avant d'aborder un autre problème clef, la recherche du *plus court chemin* entre deux nœuds d'un graphe ou digraphe.

7.1 Notions de base

Definition 7.1 Un graphe $G = (V, E)$ est défini par un ensemble V de *noeuds* ou *sommets* ($V \neq \emptyset$) et par un ensemble E d'*arêtes* qui sont des sous-ensembles de V qui contiennent deux sommets $e = \{i, j\} \in E$. On écrit $e = (i, j)$.

On suppose que $|V|$ est fini.

Definition 7.2 Un arête (i, i) est une *boucle*.

Definition 7.3 Un graphe sans boucles et avec au plus une arête entre chaque paire de sommets est un graphe *simple*. Sinon, c'est un *multigraphe*.

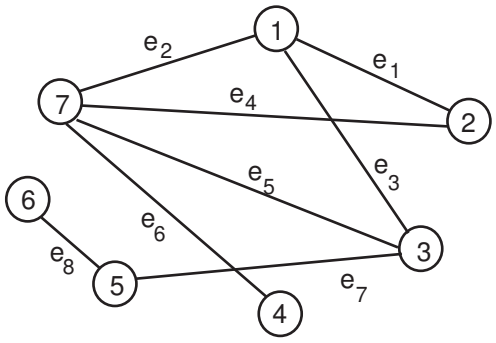


Figure 7.1: Exemple d'un graphe

Definition 7.4 La *matrice d'incidence nœuds-arêtes* est une matrice 0-1 dont les lignes représentent les sommets et les colonnes représentent les arêtes. L'élément a_{ij} de la matrice vaut 1 si l'arête e_j est incidente au sommet i, et $a_{ij} = 0$ sinon. (S'il y a des boucles $e_j = (i, i)$, on obtient une matrice 0,1,2 avec $a_{ij} = 2$.)

Example 7.1 *Matrice d'incidence :*
La matrice d'incidence du graphe simple de la Figure 4.1 est :

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\ \left(\begin{array}{cccccccc} 1 & 1 & 1 & & & & & \\ 1 & & & 1 & & & & \\ & & 1 & & 1 & & 1 & \\ & & & & & 1 & & \\ & & & & & & 1 & 1 \\ & & & & & & & 1 \\ & 1 & & 1 & 1 & 1 & & \end{array} \right) \end{array}$$

Definition 7.5 Le *degré* d'un nœud $i \in V$ est le nombre d'arêtes incidentes à i.

Definition 7.6 Un *parcours* (walk) est une séquence finie

$$W = v_0 e_1 v_1 e_2 v_2 e_3 \ \ldots \ e_k v_k \ \ (k \geq 0)$$

commençant et se terminant par un nœud et dans laquelle nœuds et arêtes se succèdent de telle façon que pour $i = 1, 2, \ldots, k$, les nœuds qui terminent l'arête e_i soient précisément v_{i-1} et v_i.

En pratique, dans un graphe simple, les parcours sont représentés uniquement par la suite des v_i : $v_0 v_1 \dots v_k$

Definition 7.7 Un *chemin* (path) est un parcours où tous les nœuds $v_0 \dots v_k$ sont différents. (Dans ce cas, les arêtes sont elles aussi distinctes).

Definition 7.8 Un *cycle* est un parcours où $v_0 = v_k$ et tous les autres nœuds doivent être distincts.

> **N.B.**
> Malheureusement, la terminologie utilisée n'est pas la même chez tous les auteurs. Par exemple, un parcours est souvent appelé une chaîne, un chemin est appelé une chaîne élémentaire, un parcours orienté est appelé un chemin. Pour éviter toute confusion nous nous tiendrons strictement aux définitions introduites ci-dessus.

Example 7.2 *Un multigraphe*
Voir Figure 4.2.

Parcours	:	2*e*4*i*5*h*4*i*5		
Chemin	:	2*b*1*c*5	=	*215*
Cycle	:	4*e*2*d*3*g*4	=	[4 2 3 4]

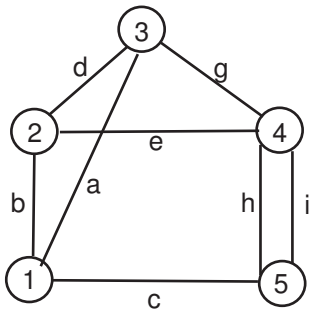


Figure 7.2: Un cycle est un parcours; un chemin est un parcours

Dans la suite, nous travaillerons surtout avec des graphes simples.

Proposition 7.1 *S'il existe un parcours $v_0, v_1, \dots, v_k$ de v_0 à v_k, il existe un chemin $v_{j_1}, v_{j_2}, \dots, v_{j_r}$ de v_0 à v_k avec $0 = j_1 < j_2 < \dots < j_r = k$.*

Preuve. Soit v_r le premier noeud tel que $v_i = v_r$ avec $0 \leq i < r$. Alors $v_0, v_1, \dots v_i, v_{i+1}, \dots, v_k$ est un parcours de 0 à k qui contient moins de noeuds. En répétant un maximum de k fois, on obtient un parcours où chaque noeud est distinct. ∎

7.2 Connexité

Definition 7.9 Un graphe est dit *connexe* si toute paire de nœuds est reliée par un chemin.

Nous avons immédiatement la propriété suivante :

Proposition 7.2 *Soit un graphe $G = (V, E)$. S'il existe un ensemble W de nœuds avec $\emptyset \subset W \subset V$ tel qu'il n'y a aucune arête (i, j) avec $i \in W$ et $j \in V \setminus W$, alors G n'est pas connexe.*

Preuve. Nous faisons la preuve par contradiction. Soit $u \in W$ et $v \in W \setminus V$. Si le graphe est connexe, il existe un chemin $u = v_0 e_1 v_1 e_2 \dots e_k v_k = v$ de u à v. Soit v_r le premier nœud sur le chemin à partir de u contenu dans $V \setminus W$. Alors $v_{r-1} \in W$ et $e_r \equiv (v_{r-1}, v_r) \in E$, une contradiction. Donc G n'est pas connexe. ∎

Dans la Figure 7.3 nous montrons des graphes connexes et non connexes.

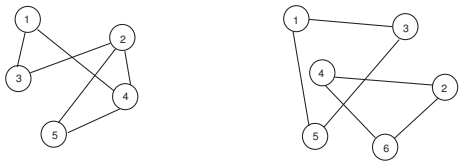


Figure 7.3:

La première question à poser quand on doit examiner un graphe est de savoir s'il est connexe ou non. Avec ce but en vue, considérons l'algorithme suivant :

Algorithme : Test de connexité

$$G = (V, E) \text{ un graphe (non trivial } : |V| \geq 2);$$

$$E' = E \text{ et } F = \emptyset$$

(i) Prendre une arête $e \in E'$, ajouter e à F, enlever e de E'

(ii) Pour toute arête $e \in E'$ qui est incidente à au moins un nœud de $V(F)$ ($V(F)$ est l'ensemble des nœuds atteints par les arêtes dans F), ajouter e à F et enlever e de E'.

(iii) Répéter (ii) jusqu'à ce que E' ne contienne aucune arête avec la propriété désirée (c.à.d. on ne peut plus décroitre $|E'|$)

L'algorithme se termine avec $F = E$ et $V = V(F)$ **ssi** G est connexe.

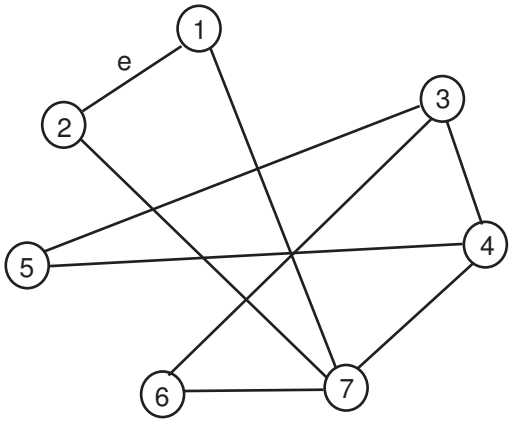


Figure 7.4:

Example 7.3 *Le graphe (a) de la figure 7.4 ci-dessous est-il connexe.?*

Si on applique l'algorithme à partir de l'arête e_1 *on obtient successivement pour* E' *(trait continu) et* F *(trait pointillé) :*

F	$V(F)$
(12)	$\{1,2\}$
$+(17),(27)$	$\{1,2,7\}$
$+(47),(67)$	$\{1,2,4,6,7\}$
$+(45),(36)$	$\{1,2,3,4,5,6,7\}$
$+(35)$	$\{1,2,3,4,5,6,7\}$
$=E$	

Parce que $V = V(F)$ *à la fin, le graphe est connexe.*

Theorem 7.3 *A la fin de l'algorithme*

(a) Le graphe $(V(F), F)$ *est connexe.*

(b) G *est connexe ssi* $V(F) = V$*.*

Preuve

(a) Par induction. Après l'étape (i) de l'algorithme $F = \{e\}$, $V(F) = \{i, j\}$ où $e = (i, j)$, et $(V(F), F)$ est connexe. Supposons que $(V(F), F)$ est connexe après $k = 1, \ldots, r-1$ iterations.

Soit V^r, F^r les nœuds et arêtes ajoutés au $(r)^e$ passage. Si $p \in V^r$, il existe $f = (p, q) \in F^r$ avec $q \in V(F)$. Parce qu'il y a un chemin entre chaque nœud de $V(F)$ et q, il y a un chemin entre chaque nœud de $V(F)$ et p.

Si $p, p' \in V^r$, on vient de montrer qu'il existe un chemin entre $q \in V(F)$ et p'. En ajoutant l'arête f, on obtient un chemin entre p et p'. Donc l'hypothèse d'induction est vérifiée pour $k = r$.

(b) A la fin de l'algorithme, il n'y a aucune arête entre un nœud de $V(F)$ et un nœud de $V \setminus V(F)$. Si $V(F) = V$, nécessairement $F = E$, et par (a), G est connexe. D'autre part si $V \setminus V(F) \neq \emptyset$, le graphe n'est pas connexe par la Proposition 2.

Corollaire. Un graphe connexe d'ordre n ($|V| = n$) doit avoir au moins $n - 1$ arêtes.
Preuve du corollaire. Il suffit d'utiliser l'algorithme et une simple induction :

- **en (i)** $|F| = 1$ et $|V(F)| = 2$, donc $|F| \geq |V(F)| - 1$ *(induction)*
- **en (ii)** à l'itération t, $|F|$ augmente de $|F^t|$ et $|V(F)|$ augmente d'au plus $|F^t|$. Cela implique que $|F| \geq |V(F)| - 1$ reste toujours vrai.
- A la fin de l'algorithme $V = V(F)$ et $|E| = |F| \geq |V(F)| - 1 = n - 1$ ∎

Definition 7.10 Soit $G = (V, E)$, et V^t, E^t pour $t = 1, \ldots, k$ une partition de V et E réspectivement, tel que $G^t = (V^t, E^t)$ est maximal connexe, alors les graphes $G^t = (V^t, E^t)$ sont les *composantes connexes* de G.

7.3 Coupe (edge cut set) et Cocycle

A modifier!

Soit $G = (V, E)$ un graphe.

Definition 7.11 Pour $\emptyset \subset W \subset V$ on définit $\delta(W)$ comme l'ensemble des arêtes qui ont une extrémité dans W et l'autre extrémité dans $V \setminus W$. $\delta(W)$ est appelé *coupe* ("edge cut-set" ou "cut-set") parce que même si G est connexe, $G' = \Big(V, E \setminus \delta(W)\Big)$ n'est plus connexe.

Definition 7.12 Un *cocycle* est une coupe minimale (non vide) c.à.d. une coupe dont on ne peut retirer aucune arête de la coupe sans perdre la propriété de séparer (couper).

Example 7.4 *Voir la Figure 7.5.*

$\delta(\{1,2,3\})$	$= \{(1,4),(2,4),(2,6),(3,5),(3,6)\}$	*est une coupe*
$\delta(\{1,2,3,6\})$	$= \{(1,4),(2,4),(3,5)\}$	*est un cocycle*
$\delta(\{6\})$	$= \{(2,6),(3,6)\}$	*est un cocycle*
$\delta(\{4,5\})$	$= \{(1,4),(2,4),(3,5)\}$	*est un cocycle.*

On voit que la coupe $\delta(\{1,2,3\})$ contient 2 cocycles $\delta(\{6\})$ et $\delta(\{4,5\})$.

Dans la Proposition 7.2, nous avons montré que si G a une coupe $\delta(W) = \emptyset$, G n'est pas connexe. Maintenant, nous montrons l'inverse.

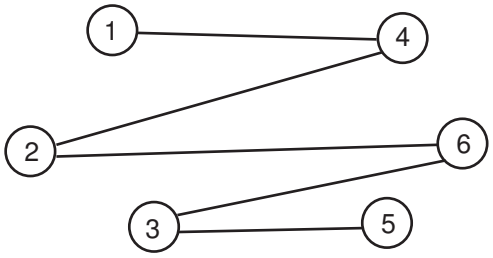


Figure 7.5:

Proposition 7.4 *Soit $G = (V, E)$ un graphe non trivial.*

$$\text{Si } \delta(W) \neq \emptyset \ \forall \ \emptyset \subset W \subset V, \quad \text{alors } G \text{ est connexe.}$$

Preuve

Soient 2 nœuds u, v quelconques.

On construit un sous-graphe $G' = (U, F)$ de la façon itérative suivante :

(a) $U = \{u\}$ et $F = \emptyset$.

(b) Prendre une arête de $\delta(U)$, soit xy, $x \in U$ et $y \notin U$,
ajouter xy à F et y à U

(c) Si $y = v$, stop
Si $y \neq v$, continuer en (b)

Parce que $\delta(U) \neq \emptyset$, l'étape (b) est toujours réalisable et la procédure doit se terminer en (c) avec $y = v$. Alors, il existe un chemin de u à v, et G est connexe.
∎

7.4 Arbres

Definition 7.13 Un graphe qui ne contient pas de cycle est une *forêt*.

Definition 7.14 Un *arbre* est une forêt connexe.

Le théorème suivant donne les principales caractérisations d'arbres.

Theorem 7.5 *Pour un graphe simple $G = (V, E)$ d'ordre $n = |V|$, les affirmations suivantes sont équivalentes et caractérisent G comme un arbre.*

1. *G est connexe et comporte $n - 1$ arêtes.*
2. *G comporte $n - 1$ arêtes et ne comporte pas de cycle. (c.à.d. une forêt à $n - 1$ arêtes)*

Figure 7.6:

3. *G est connexe et ne comporte pas de cycle.*
4. *G est connexe de façon minimale, c.à.d. tout arête de G est une coupe (si on enlève n'importe quelle arête, G n'est plus connexe)*
5. $\forall$ *paire de nœuds* u, v , $\exists!$ *chemin* $u \to v$
6. *G ne comporte pas de cycle et si on ajoute une arête on crée un et un seul cycle.*

Preuve

(1) $\Rightarrow$ (2)

On sait (par le corollaire du Théorème 7.3) qu'un graphe connexe comporte au moins $n-1$ arêtes. S'il en comporte exactement $n-1$, on doit avoir à la fin de l'algorithme qui vérifie la connexité $|F| = |V(F)| - 1$. Cela veut dire qu'à chaque étape de l'algorithme, lorsque $|F|$ augmente de 1, $|V(F)|$ doit aussi augmenter de 1. Ce ne serait pas le cas si G contenait un cycle.

(2) $\Rightarrow$ (3)

Si le graphe ne comporte pas de cycle, on doit toujours avoir $|F| = |V(F)| - 1$ au cours de l'algorithme qui teste la connexité. Si le graphe n'est pas connexe, on peut toujours utiliser cet algorithme sur chaque composante connexe du graphe.

Soit

$i = 1 \dots k$: $k \geq 2$ est le nombre de composantes connexes;

e_i et v_i le nombre d'arcs et de sommets de la composante i.

Par l'algorithme on doit avoir $e_i = v_i - 1$

Mais

$$\sum_{i=1}^{k} e_i = \sum_{i=1}^{k} v_i - k$$

$$n - 1 = n - k$$

avec $k \geq 2$ ce qui est impossible.

Le graphe G doit donc être connexe ($k = 1$)

(3) $\Rightarrow$ (1)
S'il est connexe sans cycle, on doit avoir $|E| = |V| - 1$ par l'algorithme qui teste la connexité car il se termine avec $E = F$ et $V = V(F)$

(3) $\Rightarrow$ (6)
Soit $e = (u, v)$ l'arête qui est ajoutée. Parce que G est connexe, il existe un chemin $u = v_0, v, \ldots, v_k = v$ de u à v. En ajoutant e, on obtient un cycle. Supposons maintenant qu'on crée deux cycles distincts $u, v_1, \ldots, v_{k-1}, v, u$ et
$u, w_1, \ldots, w_{r-1}, v, u$. Maintenant $u, v_1, \ldots, v_{k-1}, v, w_{r-1} \ldots, w_1, u$ est un parcours de u à u qui ne contient pas e. Par le même argument que dans la Proposition 1, il en suit qu'il y a un cycle de u à u qui ne contient pas e. Ceci contredit le fait que G est sans cycle.

(6) $\Rightarrow$ (4)
Si G n'est pas connexe, alors il existe une paire a, b de sommets (non connectés) pour lesquels l'ajout d'une arête (a, b) ne crée pas de cycle. Donc G est connexe.

Ceci établit (3) qui est déjà montré équivalent à (1). Donc G doit avoir $n - 1$ arêtes (si n est le nombre de sommets).

Si on enlève une arête, comme le graphe est toujours sans cycle, il ne peut plus être connexe [car connexe et sans cycle entraîne $n - 1$ arêtes]

(4) $\Rightarrow$ (5)
G connexe $\Rightarrow \forall u, v \in V \; \exists$ chemin $u \rightarrow v$

Si pour une paire $u, v, \in V$ il existait plusieurs chemins entre u et v, la suppression d'un arc d'un des chemins ne déconnecterait pas le graphe.

(5) $\Rightarrow$ (3)
G connexe car $\forall u, v \in V$ il existe un chemin $u \rightarrow v$.

Si G admettait un cycle, alors il existerait (au moins) une paire de sommets reliés par 2 chemins distincts ∎

Corollaire
Un graphe est connexe **ssi** il contient un arbre.
Preuve du corollaire
Un arbre T est connexe, donc tout graphe qui contient l'arbre T sera connexe. Inversément, si G est connexe, alors on peut obtenir à partir de G (en enlevant des arcs) un graphe G' connexe de façon minimale.

Le théorème précédent nous dit que G' est un arbre. ∎

7.5 Arbre de poids minimum

A chaque arête e d'un graphe connexe G, on associe un poids (coût) c_e. Le problème posé est de construire un arbre dans le graphe G dont la somme des poids des arêtes soit minimum.

7.5.1 Algorithme 1

Glouton (Greedy) (Kruskal)

(i) Classer les arêtes par ordre croissant de poids:

$$c_{e_1} \leq c_{e_2} \leq \ldots \leq c_{e_m}.$$

(ii) Soit $E' = E$ et $T = \emptyset$.

(iii) Prendre l'arête de E' qui a le poids le plus petit :
soit e

- Enlever e de E'.
- Si $T \cup \{e\}$ ne comporte pas de cycle, alors on ajoute e à T.
  Sinon on ne met pas e dans T.

(iv) On continue (iii) jusqu'à ce que $|T| = n - 1$.

Proposition 7.6 *L'algorithme 1 résout le problème posé.*
L'algorithme est **correct**.

Preuve Supposons (pour la simplicité de la preuve) que

$$c_{e_1} < c_{e_2} < \ldots < c_{e_m}.$$

Soit $f_1 f_2 \ldots f_{n-1}$ les arêtes de la "*solution glouton*", avec $c_{f_1} < c_{f_2} < c_{f_3} < \ldots$
Soit $e_1 e_2 \ldots e_{n-1}$ les arêtes de la *solution optimale* avec $c_{e_1} < c_{e_2} < c_{e_3} < \ldots$
Si la "solution glouton" n'est pas la solution optimale, alors il doit exister r tel que :

$$e_1 = f_1 \; ; \; e_2 = f_2 \; ; \; \ldots \; e_{r-1} = f_{r-1} \; ; \; e_r \neq f_r.$$

Assertion 1

$$c_{f_r} < c_{e_r}$$

sinon e_r aurait été choisi à la place de f_r dans l'algorithme "glouton".
Assertion 2

$$\{e_1, e_2, \ldots, e_{n-1}, f_r\}$$

contient exactement un cycle C car $\{e_1, \ldots, e_{n-1}\}$ est un arbre.
Assertion 3
Au moins un des arêtes $e_r \ldots e_{n-1}$ est dans le cycle C car $\{e_1, \ldots, e_{r-1}, f_r\}$ ne

contient pas de cycle.
Soit e_k une arête dans le cycle C avec $k \geq r$
Assertion 4

$$\{e_1, e_2, \ldots e_{n-1}, f_r\} \backslash \{e_k\}$$

est un arbre car on retire une arête du seul cycle de $\{e_1, e_2, \ldots, e_{n-1}, f_r\}$
Assertion 5

$$\text{Comme } c_{f_r} < c_{e_r} < c_{e_k},$$

on a trouvé un arbre de poids moindre que $\{e_1 \ldots e_{n-1}\}$, et on a une contradiction.
Donc il n'existe pas de r tel que $e_r \neq f_r$,et la "solution glouton" doit être optimale. ∎

7.5.2 Présentation formelle

On peut présenter l'algorithme de façon plus formelle :

Soit T l'ensemble d'arêtes formant une forêt (donnant l'arbre optimal à la fin de l'algorithme) dont les composantes connexes sont $F_1 \ldots F_r$.

Soit VT un ensemble comprenant les ensembles des nœuds $V(F_1), \ldots V(F_r)$. $V(F)$ est l'ensemble des sommets touchant au moins une arête de F, et VT est donc une partition de l'ensemble V des sommets.

Algorithme 1'
$T \leftarrow \emptyset$
$VT \leftarrow \{1\}, \{2\}, \ldots, \{n\}$
Q : structure contenant les arêtes de E
Tant que $|VT| > 1$, do
Begin
  Trouver l'arête (v, w) de Q de moindre coût.
  Enlever (v, w) de Q
  Si v et w sont dans des ensembles $V(F_i)$ et $V(F_j)$ différents
  Alors
    Enlever $V(F_i)$ et $V(F_j)$ de VT
    Ajouter $V(F_i) \cup V(F_j)$ dans VT
    Ajouter (v, w) à T
    End
End

7.5.3 Temps d'exécution

La question suivante à se poser est celle du temps d'exécution nécessaire à cet algorithme.

A chaque itération, on enlève de Q l'arête de moindre poids. Il faut donc trier les arêtes de E par ordre croissant de poids. La quantité de travail nécessaire pour cette étape (heapsort par exemple) est de l'ordre de $O(|E| \log |E|)$.

Ensuite, à chaque itération, on doit trouver quels ensembles $V(F_i)$ contiennent les sommets v et w. Cela peut se faire très facilement à l'aide de pointeurs. Enfin, prendre l'union de deux ensembles $V(F_i)$ et $V(F_j)$ ne nécessite que de changer les pointeurs associés aux nœuds d'un des ensembles. Il y a au maximum $|E|$ itérations et au maximum $|V|$ unions d'ensembles à prendre, et il est possible de faire ces opérations dans un temps inférieur à $O(|E| \log |E|)$.

Donc le temps d'exécution est dominé par le temps nécessaire au tri des arêtes, et l'algorithme est de l'ordre de $O(|E| \log |E|)$

7.5.4 Algorithme de PRIM

L'étape suivante est de songer à d'éventuelles améliorations de cet algorithme.

L'idée (due à Prim) est de construire un ensemble de nœuds connexes, de façon optimale et de plus en plus grand. Soit P cet ensemble de nœuds et $T = V \backslash P$:

(i) **Initialisation** :

$G = (V, E)$ et c_{ij} (poids sur les arêtes) sont données.

Soit $P = \{1\}$ et $T = \{2, 3, \ldots, n\}$ $S = \emptyset$

Soit $i(j) = 1$ et $u_j = c_{1j}$ pour $j = 2, 3, \ldots, n$ $(j \in T)$

(ii) **Addition d'arête à l'arbre** :

Trouver

$$k \in T \text{ tel que } u_k = \min_{j \in T} \{u_j\}.$$

Soit

$$T \leftarrow T \backslash \{k\} \; ; \; P \leftarrow P \cup \{k\}$$

$$S \leftarrow S \cup \{(i(k), k)\}$$

Si $T = \emptyset$, STOP . S est l'arbre optimal.

(iii) **Révision des labels**

$$\forall j \in T \text{ si } c_{kj} < u_j \text{ alors } u_j \leftarrow c_{kj} \text{ et } i(j) = k$$

(iv) **goto (ii)**

La preuve de validité de l'algorithme peut être faite par un argument similaire à celui utilisé pour l'algorithme glouton.

La complexité de l'algorithme est $O(|V|^2)$ (car il y a $O(|V|)$ itérations où $O(|V|)$ opérations de comparaison sont requises).

On voit donc que pour des graphes denses ($|E| \approx |V|^2$) ; ce nouvel algorithme est beaucoup plus performant.

Exercice Prouvez la validité de cet algorithme.

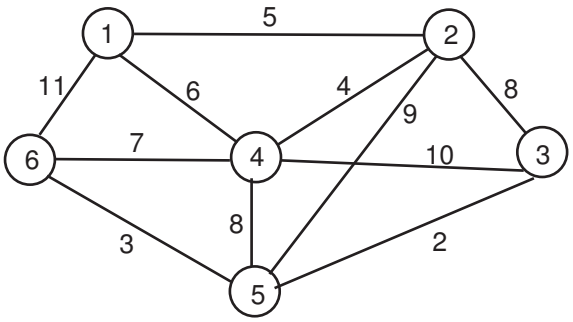


Figure 7.7:

Example 7.5 *Nous appliquons l'algorithme glouton au graphe de la Figure 7.7.*
$c_{35} < c_{56} < c_{24} < c_{12} < c_{14} < c_{46} < c_{23} = c_{45} < c_{25} < c_{34} < c_{16}$
$(34) \in T$ *Composantes connexes 1,2,35,4,6*
$(56) \in T$ *Composantes connexes 1,2,356,4*
$(24) \in T$ *Composantes connexes 1,24,356*
$(12) \in T$ *Composantes connexes 124,356*
$(14) \notin T$ *Composantes connexes 124,356*
$(46) \in T$ *Composantes connexes 123456. Stop.*

Ensuite nous montrons les étapes de l'algorithme de Prim.

(i) $P = \{6\}$ *et* $T = \{1, 2, 3, 4, 5\}$ *(le nœud initial de* P *est quelconque)*
$u_1 = 11, u_4 = 7, u_5 = 3, u_i = \inf$ *autrement.*

(ii) k *est le nœud de* T *le plus proche d'un nœud (quelconque) de* P*.*
$k = 5 et\ i(5) = 6; P = \{5, 6\}$ *et* $T = \{1, 2, 3, 4\}$

(iii) $u_1 = 11\ ;\ u_2 = 9\ ;\ u_3 = 2\ ;\ u_4 = 7$

(ii) $k = 3$ *et* $i(3) = 5, P = \{3, 5, 6\}$ *et* $T = \{1, 2, 4\}$

(iii) $u_1 = 11\ ;\ u_2 = 8\ ;\ u_4 = 7$

(ii) $k = 4$ *et* $i(4) = 6, P = \{3, 4, 5, 6\}\ ;\ T = \{1, 2\}$

(iii) $u_1 = 6; u_2 = 4$

(ii) $k = 2$ *et* $i(2) = 4, P = \{2, 3, 4, 5, 6\}\ ;\ T = \{1\}$

(iii) $u_1 = 5\ ;\ i(1) = 2$

(ii) On a une solution optimale.

Chapter 8

Chemins

8.1 Graphes et Isomorphisme

Une question naturelle à poser: Est-ce que deux graphes $G = (V, E)$ et $G' = (V', E')$ sont les mêmes?

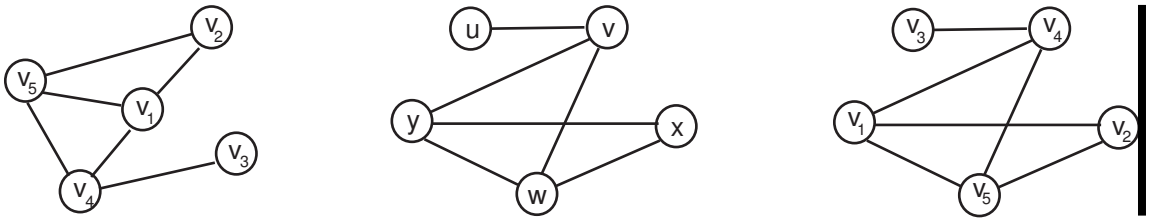


Figure 8.1: Exemple de graphes identiques et isomorphes

Considérons les trois graphes de la figure 8.1.

Les graphes G_1 et G_3 sont identiques.

Les graphes G_2 et G_3 ont la même forme. Ils ne sont pas identiques, mais **isomorphes**.

Definition 8.1 G_1 et G_2 sont **isomorphes** s'il existe une bijection

$$\Phi = V_1 \to V_2$$

tel que

$$(i, j) \in E_1 \textbf{ ssi } (\Phi(i), \Phi(j)) \in E_2.$$

On note $G_1 \cong G_2$.

Dans la Figure 8.1, la fonction Φ définie par

$$\Phi(u) = v_3,\ \Phi(v) = v_4,\ \ \Phi(x) = v_2,\ \ \Phi(w) = v_5 \text{ et } \Phi(y) = v_1.$$

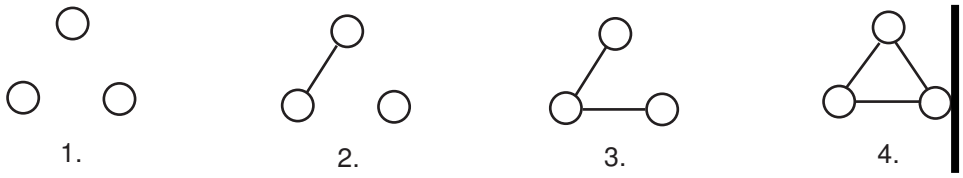


Figure 8.2: Graphes sur 3 sommets

montre que G_2 et G_3 sont isomorphes.

Les différents graphes sur 3 sommets sont donnés à la Figure 8.2.

Un problème important est de voir si deux graphes sont **isomorphes**. Ce problème ne sera pas abordé en détail ici, mais il est évident que pour bien l'aborder, et tout autre problème de calcul avec des graphes, il faut choisir une représentation claire.

8.2 La matrice d'adjacence de G.

Nous avons déjà vu la matrice d'incidence d'un multigraphe avec $A = (a_{ij})$, où

$$a_{ij} = 1 \text{ si } i \text{ touche l'arête } e_j.$$

$$a_{ij} = 2 \text{ si } e_j = (i, i),$$

$$\text{et } a_{ij} = 0 \text{ autrement.}$$

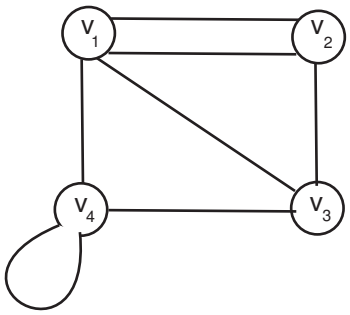


Figure 8.3:

Cette matrice a $n = |V|$ lignes et $m = |E|$ colonnes. Une façon plus compacte de répresenter G est d'utiliser *une matrice d'adjacence* qui a pour dimensions $|V| \times |V|$, avec

$$m_{ij} = \text{ nombre d'arêtes entre } i \text{ et } j.$$

Example 8.1 *Matrice d'adjacence.*
Ici on montre la matrice obtenue pour le multigraphe de la Figure 8.3.

$$M(G) = \begin{array}{c|cccc} & v_1 & v_2 & v_3 & v_4 \\ \hline v_1 & 0 & 2 & 1 & 1 \\ v_2 & & 0 & 1 & 0 \\ v_3 & & & 0 & 1 \\ v_4 & & & & 2 \end{array}$$

La matrice M est souvent plus petite que A, et donc on représente souvent le graphe par $M(G)$.

Si le graphe est simple et n'a pas beaucoup d'arêtes on utilise souvent des listes par nœud.

$N(i)$ est l'ensemble de noeuds voisins du noeud i.

$$\begin{array}{lcll} \text{Pour le noeud 1} & : & V(1) \subseteq V & = & \{j : \text{il existe une arête } (1,j)\} \\ \text{pour le noeud 2} & : & V(2) \subseteq V & = & \{j : \text{il existe une arête } (2,j)\} \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{array}$$

Example 8.2 *Représentation par listes de voisins.*
Les nœuds de la Figure 8.4 sont listés de la façon suivante :

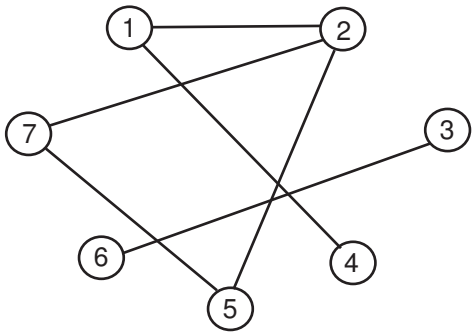


Figure 8.4:

$$\begin{array}{rcl} \textit{voisinage de } \textit{1} & : & \{2,4\} \\ \text{"} \quad \textit{2} & : & \{1,5,7\} \\ \text{"} \quad \textit{3} & : & \{6\} \\ \text{"} \quad \textit{4} & : & \{1\} \\ \text{"} \quad \textit{5} & : & \{2,7\} \\ \text{"} \quad \textit{6} & : & \{3\} \\ \text{"} \quad \textit{7} & : & \{2,5\} \end{array}$$

8.3 Le degré des nœuds

Definition 8.2 Le *degré* $d(v)$ d'un nœud $v \in V$ est le nombre d'arêtes incidentes à v. Les arêtes (v, v) comptent pour deux dans l'expression du degré $d(v)$.

Proposition 8.1

$$\sum_{v \in V} d(v) = 2|E|$$

Preuve Considérer la matrice d'incidence A. La somme de chaque ligne est le degré de v. La somme de chaque colonne $= 2$.
Donc

$$\sum_{v \in V} d(v) = 2|E|.$$

∎

Corollaire Le nombre de nœuds de degré impair est pair.
Preuve du corollaire
Soient V_0 les nœuds de degré pair et V_1 les nœuds de degré impair :

$$\sum_{v \in V_0} d(v) + \sum_{v \in V_1} d(v) = 2|E|.$$

Donc $\sum_{v \in V_1} d(v)$ est paire.
Donc $|V_1|$ est paire.

8.4 Chemins, parcours et cycles

Revenons aux parcours et aux chemins.

Proposition 8.2 *Le nombre de (v_i, v_j) parcours de longueur k est le coefficient (i, j) de M^k.*

Pour le graphe de la figure 8.4 on obtient :

$$M = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{array} \begin{array}{c} \begin{array}{ccccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{array} \\ \begin{pmatrix} & 1 & & 1 & & & \\ 1 & & & & 1 & & 1 \\ & & & & & 1 & \\ 1 & & & & & & \\ & 1 & & & & & 1 \\ & & 1 & & & & \\ & 1 & & & 1 & & \end{pmatrix} \end{array}$$

$M^2 = (m_{ij}^2)$: $m_{ij}^2 = \sum_k m_{ik} m_{kj}$ = nombre de parcours de longueur 2.

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

Definition 8.3 Un graphe est **bipartite** si $V = V_1 \cup V_2$ et chaque arête (i, j) a $i \in V_1$ et $j \in V_2$ (voir Figure 8.5).

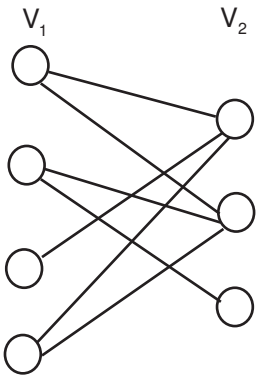


Figure 8.5: Exemple d'un graphe bipartite

Proposition 8.3 *Un graphe est bipartite* **ssi** *il n'y a pas de cycle de longueur impaire.*

Preuve A refaire!

- $\Rightarrow$ Si G est bipartite, et $C = v_0, e_1, v_1, \ldots, v_{k-1}, e_k, v_k, e_{k+1}, v_0$ est un cycle, $v_{2i} \in V_1,\ v_{2i+1} \in V_2$. Donc $k = 2i + 1$ et le nombre d'arêtes dans le cycle est $k + 1$ qui est pair.
- $\Leftarrow$ Supposons que G est connexe sans cycle impair.

  Choisir $u \in V$

  Soient
  $$X = \{x \in V \ : \ \text{dist}(u, x) \text{ est paire } \}$$
  et
  $$Y = \{x \in V \ : \ \text{dist}(u, x) \text{ est impaire } \}$$

  Comme exercice, montrer que (X, Y) est une partition de V.

8.5 Le Problème du Plus Court Chemin d'un Graphe

Soit $d(u, v)$ = nombre minimum d'arêtes sur un chemin de u à v.

Comment calculer $d(1, v)\ \forall v \in V \backslash \{1\}$?

A distance 0 : $S_0 = \{1\}$

A distance 1 : $S_1 = V(1)$, les voisins du nœud 1

A distance 2 : $S_2 = \bigcup_{j \in S_1} V(j) \backslash (S_0 \cup S_1)$

A distance 3 : $S_3 = \bigcup_{j \in S_2} V(j) \backslash (S_0 \cup S_1 \cup S_2)$

...

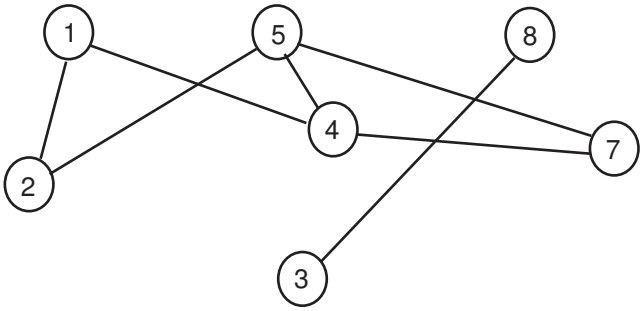


Figure 8.6:

Example 8.3
Considérons le graphe de le Figure 8.6. En utilisant les listes de voisins données à l'exemple 3, on obtient facilement :

$S_0 = \{1\}$
$S_1 = V(1) = \{2, 4\}$
$S_2 = V(2) \cup V(4) \setminus (S_0 \cup S_1) = \{5, 7\}$
$S_3 = \emptyset$

Avant d'aborder le problème plus général où des coûts sont associés à chaque arête, il est opportun d'introduire les graphes orientés. Pour ces graphes aussi, naturellement, se posent les questions d'existence de chemins.

8.6 Digraphes

La plupart des définitions sont des extensions naturelles de celles des graphes.

8.7 Chemin de Coût Minimum d'un Digraphe

Le problème Pour un digraphe $D = (V, A)$ avec des coûts c_{ij} pour chaque arc (i, j), trouver le chemin de s à t de coût minimum.

Observons d'abord que le problème de trouver le chemin orienté de s à t avec le nombre minimum d'arcs peut être résolu de la même façon que le cas non orienté avec les listes de voisins $\delta^+(j)$, (voir l'exemple 2). C'est aussi un cas particulier du problème de chemin à coût minimum avec $c_{ij} = 1 \ \forall \ (i, j) \in A$.

De plus, le problème de chemin à coût minimum dans un graphe $G = (V, E)$ est aussi un cas particulier du problème sur un digraphe $D = (V, A)$. Pour tout $e = (i, j) \in E$ avec poids c_e, il suffit de prendre $A = \bigcup_{e \in E} \{(i, j) \cup (j, i)\}$ avec $c_{ij} = c_{ij} = c_e$.

L'algorithme que nous allons présenter n'est pas le plus efficace (en d'autres cours, vous entendrez parler probablement de l'algorithme de Dijkstra), mais il est à nouveau basé sur une récurrence simple, comme le problème du produit de matrices au Chapitre 3. Cette approche par récurrence s'appelle la *Programmation Dynamique*.

Maintenant prenons les coûts c_{ij}, $(i, j) \in E$. Comment trouver le chemin de coût minimum de "1" aux autres nœuds...?

Problème du Chemin de Coût Minimum pour un Digraphe $D = (V, A)$

Soit $d(1, j) =$ coût du chemin le moins coûteux de 1 à j.

Soit $D_k(j) =$ coût du chemin le moins coûteux de 1 à j avec un nombre d'arcs $\leq k$

Alors

$$D_0(1) = 0, \ \ D_0(v) = +\infty \quad v \in V \backslash \{1\}$$

et

$$D_1(j) = c_{1j}$$

pour $j \in \delta^+(1)$.

Cherchons maintenant une récurrence pour $D_k(j)$:

Il y a deux possibilités:

1. Il faut $< k$ arcs. Alors
$$D_k(j) = D_{k-1}(j).$$

2. Il faut k arcs, et i est l'avant dernier nœud. Alors
$$D_k(j) = D_{k-1}(i) + c_{ij}$$

L'algorithme

Pas 1 (Initialisation) $n = |V|$. $D_0(1) = 0$, $D_0(j) = \infty \ \ \forall j \in V \backslash \{1\}$, $k = 1$.

Pas 2 $\forall j \in V \qquad D_k(j) = \min\left\{ \min_{i:(i,j)\in A} \left[c_{ij} + D_{k-1}(i)\right],\ D_{k-1}(j) \right\}$

Pas 3

Si $D_{k-1}(j) = D_k(j)\ \ \forall j \in V$,
alors $d(1,j) = D_k(j)\ \ \forall j$.
Sinon, si $k < n,\ k \leftarrow k+1$.
Si $k = n$, le graphe contient un cycle de coût négatif.

Example 8.4 *À partir du graphe de la Figure 8.7, on établit le tableau suivant, en utilisant*

$$c_{ij} = c_{ji} = c_e$$

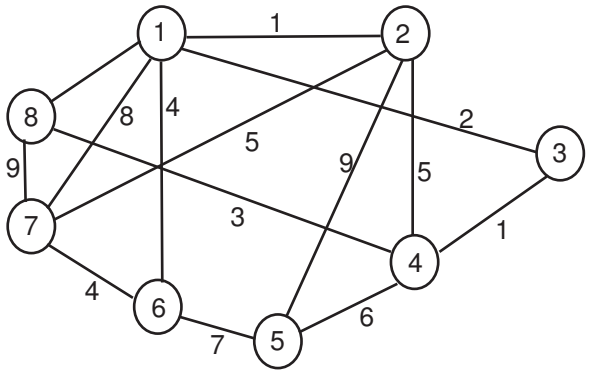


Figure 8.7:

j	$D_0(j)$	$D_1(j)$	$D_2(j)$	$D_3(j)$	$D_4(j)$
1	*0*	*0*	*0*	*0*	*0*
2	∞	*1*	*1*	*1*	*1*
3	∞	*2*	*2*	*2*	*2*
4	∞	∞	*3*	*3*	*3*
5	∞	∞	*10*	*9*	*9*
6	∞	*4*	*4*	*4*	*4*
7	∞	*8*	*6*	*6*	*6*
8	∞	*7*	*7*	*6*	*6*

Example 8.5
Sur base du digraphe de la Figure 8.8 on construit le tableau suivant :

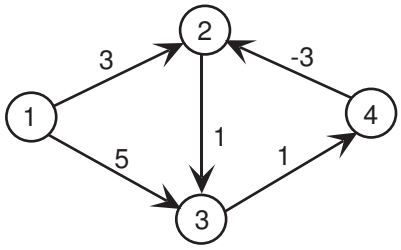


Figure 8.8: Exemple avec un cycle négatif

	nœuds			
k	**1**	**2**	**3**	**4**
0	*0*	∞	∞	∞
1	*0*	*3*	*5*	∞
2	*0*	*3*	*4*	*6*
3	*0*	*3*	*4*	*5*
4	*0*	*2*	*4*	*5*

Alors il y a un cycle négatif. On le retrouve en retraçant ses pas à partir du nœud 2, et on obtient le cycle (2,3,4,2).

Preuve que l'algorithme est correct

$D^k(j)$ = poids du parcours de 1 à j avec $\leq k$ arêtes. Si $D^n(j^*) < D^{n-1}(j^*)$ il existe un $1 - j^*$ parcours, avec n arêtes, qui a moins de poids qu'un $1 - j^*$ parcours avec un nombre d'arêtes $\leq n - 1$.

Un parcours avec n arêtes contient un chemin avec $< n - 1$ arêtes. Donc il existe un cycle. Ce cycle doit être de longueur négative.

Observations

- Les arêtes des plus courts chemins de 1 à j forment un arbre.
- Nombre de calculs = $O\left(|V|^3\right)$
- Il y a un meilleur algorithme (celui de Dijkstra) qui marche quand $c_{ij} \geq 0$. $(O(n^2))$.

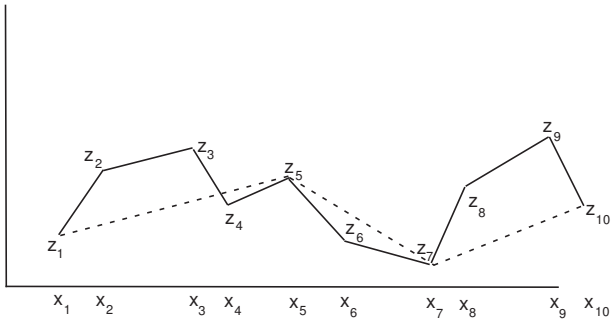
Chapter 9

Chemins et Tours

9.1 Modélisation

Nous regardons d'abord plusieurs problèmes qui peuvent être formulés et résolus comme des problèmes de plus court chemin, ou ses variantes.

Exemple 1. Approximation d'une fonction.
Une fonction $f : R^1 \to R^1$ linéaire par morceaux est donnée à travers n points $z_i = (x_i, f(x_i))$ pour $i = 1, \ldots, n$.
On veut trouver une fonction $g : R^1 \to R^1$ qui passe par $m < n$ de ces points (z_1 et z_n inclus), et qui donne la meilleure approximation possible de f.



Si m n'est pas specifié d'avance, il y a aussi bien un coût par point qu'un coût proportionel à la carré de l'erreur à chaque point z_i. Donc si g a un segment linéaire entre les points z_i et z_j, le coût de ce segment est

$$c_{ij} = \alpha + \beta \sum_{k=i}^{j} (f(x_k) - g(x_k))^2.$$

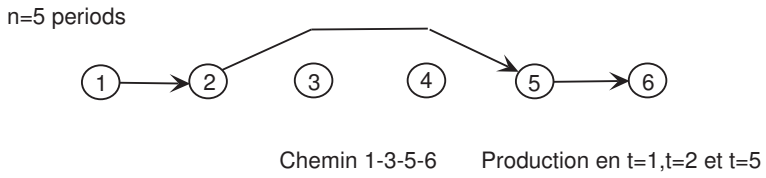
Alors le problème revient à un problème de plus court chemin sur un digraphe avec $V = \{1, \ldots, n\}$, $A = \{(i, j) : 1 \leq i < j \leq n\}$ avec c_{ij} la longueur de l'arc (i, j).

Exemple 2. Production par Lots.
Soit $d_1, \ldots, d_n$ les demandes mensuelles pour un article. Si on produit en période t, il y a un coût de lancement f_t, et un coût unitaire de production p_t. Il y a aussi un coût de stockage unitaire h_t à la fin de la chaque periode t. Trouver le plan de production de coût minimum.

Si on décide de produire en période i pour satisfaire la demande pour les périodes $i, i+1, \ldots, j-1$, le coût pour la production pendant cet intervalle est

$$c_{ij} = f_i + p_i(d_i + \ldots + d_{j-1}) + \sum_{k=i}^{j-2} h_k(d_{k+1} + \ldots + d_{j-1}).$$

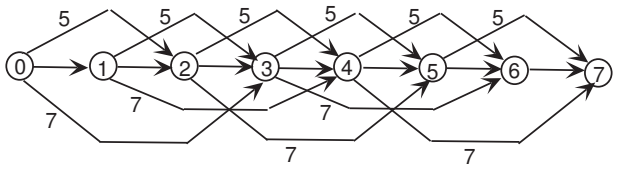
Donc il suffit de construire un digraphe avec $V = \{1, \ldots, n+1\}$ et des arcs (i, j) de longueur c_{ij}, et trouver le plus court chemin de 1 jusquà $n+1$.



Observez que nous avons fait l'hypothèse qu'il existe toujours une solution optimale dans laquelle on satisfait toujours la demande entière d'une période en une fois.

Exemple 3. Le Problème du Sac-à-Dos.
Vous partez en montagne pour quelques jours, et vous voulez que le poids de la nourriture que vous portez ne dépasse pas b kilos. n différents articles de nourriture sont proposés avec un poids a_j et une valeur nutritive c_j pour $j = 1, \ldots, n$. Trouver une solution qui maximise la valeur nutritive.

Si $n = 3$, $b = 7$, avec $(c_1, a_1) = (5, 2), (c_2, a_2) = (2, 1), (c_3, a_3) = (7, 3)$, on peut représenter le problème par un problème de plus court chemin de 0 à 7 dans le digraphe suivant. Expliquez.



Exemple 4. Circuit Négatif.
Le propriétaire d'une Smart doit décider d'un trajet pour optimiser son profit

moyen. S'il utilise l'arc (i,j) son profit est de p_{ij}, et le temps pour traverser l'arc est τ_{ij}. Il cherche un circuit C dont le profit moyen dépasse une valeur donnée:

$$\frac{\sum_{(i,j)\in C} p_{ij}}{\sum_{(i,j)\in C} \tau_{ij}} > \mu.$$

Autrement dit: Y a-t-il un circuit C avec $\sum_{(i,j)\in C} p_{ij} > \mu \sum_{(i,j)\in C} \tau_{ij}$, ou

$$\sum_{(i,j)\in C} (\mu\tau_{ij} - p_{ij}) < 0?$$

Le problème est donc de trouver s'il existe un circuit de coût négatif.

Ces quatre problèmes mènent tous à un problème de plus court chemin, ou une variante.

9.2 Ordre Topologique et Digraphes Acycliques

Definition 9.1 Un digraphe $D = (V, A)$ est *acyclique* s'il ne contient pas de dicycles.

Definition 9.2 Soit D un digraphe avec $|V| = n$ et *ordre*: $V \to \{1, \ldots, n\}$ une bijection. L'ordre est topologique si
"$ordre(i) < ordre(j)$" pour tout $(i,j) \in A$.

Dans la Figure 9.1, on montre un digraphe acyclique. Pour le premier, l'ordre indiqué n'est pas topologique. Pour le seconde on indique un ordre topologique.

Question 1. Y a-t-il des digraphes sans ordre topologique?
Réponse. Oui, chaque digraphe qui contient un dicycle.
Autrement dit,

Proposition 9.1 *Si un digraphe D a un ordre topoloqique, le digraphe est acyclique.*

Question 2. Peut-on affirmer la réciproque: chaque digraphe acyclique possède un ordre topologique?
Pour répondre, considérons les propriétés des digraphes acycliques.

Proposition 9.2 *Dans un digraphe D, si chaque noeud a au moins un arc qui entre, le digraphe contient un dicycle.*

Preuve. Par hypothèse, pour chaque $i \in V$, il existe $p(i) \in V$ tel que $(p(i), i) \in A$. Choisir arbitrairement un noeud s et définir $i_0 = s$, et $i_k = p(i_{k-1})$ pour $k = 1, 2, \ldots, n$. $i_n, i_{n-1}, \ldots, i_0$ est un parcours dirigé qui contient $n+1$ noeuds. Par le principe du tiroir, il doit y avoir répétition. Donc D contient un dicycle.

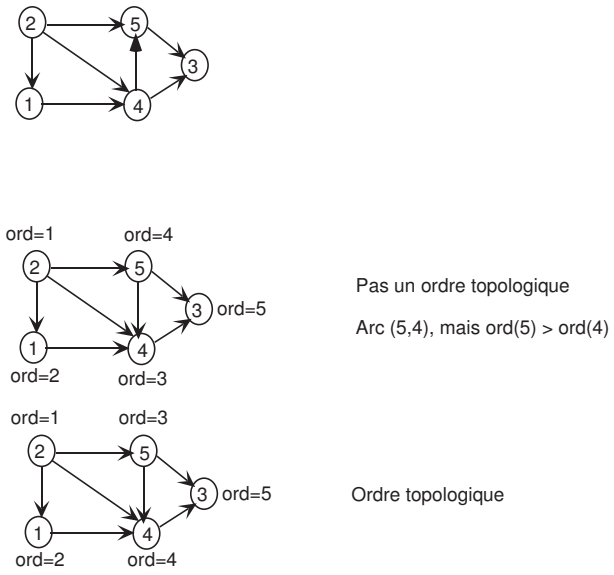
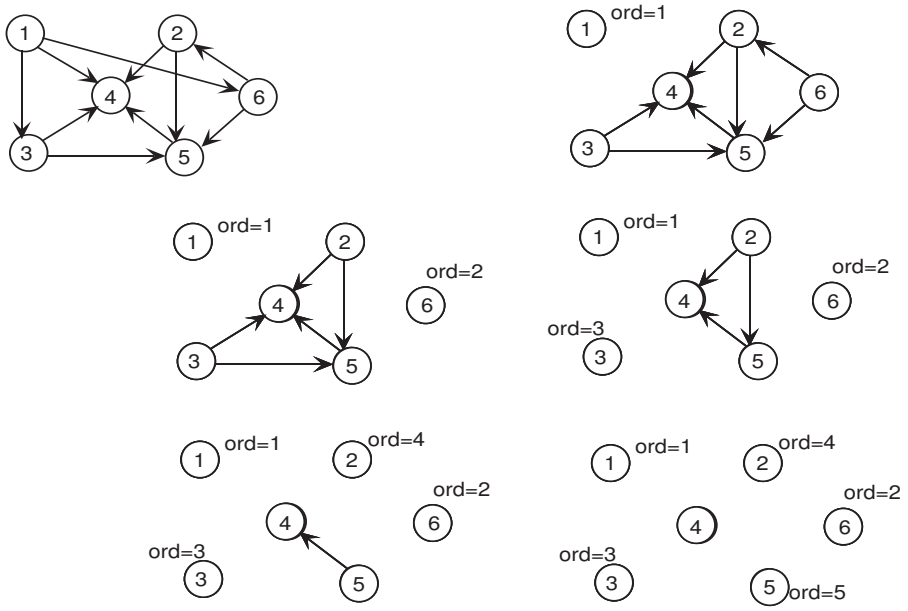


Figure 9.1: Ordre Topologique?

Corollaire 1. Dans un digraphe acyclique, il existe au moins un noeud v avec $deg^-(v) = 0$.

Corollaire 2. Dans un digraphe D avec un noeud v avec $deg^-(v) = 0$, D est acyclique ssi le digraphe D_v induit par les noeuds $V \setminus \{v\}$ est acyclique.
Preuve. Le dicycle ne peut pas passer par le noeud v.

Algorithme pour tester si D est acyclique.
1. $D^0 = D$
2. $t = 0$
3. Tant qu'il existe un noeud v avec $deg^-(v) = 0$,
4. $\quad t \leftarrow t + 1$
5. $\quad D^t = D^{t-1} \setminus \{v\}$
6. $\quad ord(v) = t$
7. Si $t = n, D$ est acyclique et ord est un ordre topologique.
8. Si $t < n$, le sous-digraphe D^t contient un dicycle.



9.3 Le Problème du Plus Court Chemin

Une propriété de base que nous avons déjà utilisé concerne l'existence d'un chemin.

Proposition 9.3 *Dans un digraphe D, s'il y a un parcours dirigé de s à t, il y a un chemin dirigé de s à t.*

Preuve. Nous donnons la construction du chemin. Soit $s = v_0, (v_0, v_1), v_1, \ldots, v_{r-1}, (v_{r-1}, v_r), v_r = t$ le parcours. Si tous les sommets $v_0, \ldots, v_r$ sont distincts, c'est un chemin. Sinon, soit v_k le premier sommet qui se répète, et $v_l = v_k$ avec $l > k$ la dernière fois que ce sommet réapparait. Alors on peut enlever le dicycle fermé $v_k, v_{k+1}, \ldots, v_l$, et ce qui reste $v_0, v_1, \ldots, v_k, v_{l+1}, \ldots, v_r$ est un parcours dirigé de s à t qui contient le sommet v_k qu'une seule fois. Et on répète un maximum de n fois.

Le Problème du Plus Court Chemin: Propriétés
Etant donné un digraphe $D = (V, A)$, un sommet $s \in V$, et une vecteur de coûts/distances c_{ij} pour $(i, j) \in A$, trouver un chemin de coût/longueur minimum de s à j pour tout $j \in V \setminus \{s\}$.

Nous voyons une variété d'algorithmes, mais il est intéressant d'aussi penser au Problème du Plus Court Parcours. En particulier, il faut considérer quand les deux problèmes sont équivalents. D'après la Proposition ci-dessus, il est crucial de savoir si D contient un dicycle de longueur négatif, ou non.

Hypothèse A.
Le digraphe ne contient aucun cycle de longueur négatif.

Proposition 9.4 *Sous l'hypothèse A, Problème du Plus Court Parcours a une solution optimale qui est un chemin.*

Preuve. En utilisant la preuve de Proposition 9.3, tous les dicycles trouvés ont une longueur nonnegative, et donc chaque parcours est au moins aussi court que le précédent. En particulier le dernier qui est un chemin doit aussi être optimal.

Proposition 9.5 *Si l'hypothèse A est satisfaite, et $v_0, v_1, \ldots, v_r$ est un plus court chemin de v_0 à v_r, alors $v_i, v_{i+1}, \ldots, v_j$ est un plus court chemin de v_i à v_j pour tout $0 \leq i < j \leq r$.*

La preuve est immédiate par contradiction.

Proposition 9.6 *Si l'hypothèse A est satisfaite, soit $D(j)$ la longueur du PCC de s à j. Alors un dichemin $P = (s = v_0, v_1, \ldots, v_r = j)$ est un plus court chemin si et seulement si $D(v_i) = D(v_{i-1}) + c_{v_{i-1},v_i}$ pour tout arc (v_{i-1}, v_i) de P.*

Corollaire. Il existe un arbre dirigé de racine s tel que le chemin unique de s à j est un plus court chemin pour tout $j \in V \setminus \{s\}$. Donc il faut seulement $n-1$ arcs pour décrire $n-1$ plus court chemins, au lieu de $O(n^2)$.

9.4 PCC dans un Digraphe Acyclique

Nous pouvons tester si un digraphe est acyclique en $O(m)$, et dans le cas positif, l'algorithme construit un ordre topologique, tel que $ord(i) < ord(j)$ pour tout $(i,j) \in A$.

Donc on peux réordonner les sommets d'après *ord* pour que chaque arc $(i,j) \in A$ satisfait $i < j$.

Algorithme
1. $d(1) = 0, d(j) = \infty$ for $j > 1$
2. for i=1,. . . ,n-1 (Selection du noeud)
3. for $j \in V^+(i)$
4. $d(j) \leftarrow \min[d(j), d(i) + c_{ij}]$. (Mise à jour des distances)
5. Mise à jour pred(j)
6. end
7. end

Theorem 9.7 *A la fin de l'algorithme, $d(j)$ est le longueur du plus court chemin de 1 à j pour tout j. La complexité de l'algorithme est $O(m)$.*

Preuve. La preuve est par induction. Soit $H(k)$ l'hypothèse que quand i=k en ligne 6, $d(j)$ est le longueur d'un PCC de 1 à j dont l'avant dernier noeud est plus petit que $k+1$.

Clairement $H(1)$ est vraie. Supposons que $H(k)$ est vraie pour $k < n-1$. Maintenant en ligne 2, $i = k+1$, et en ligne 4 $d(j) \leftarrow \min[d(j), d(k+1)+c_{k+1,j}]$ pour $j \in V^+(k+1)$. Donc $d(j)$ devient le longueur d'un PCC de 1 à j dont l'avant dernier noeud est plus petit que $k+2$. Donc $H(k+1)$ est satisfaite quand $i = k+1$ en ligne 6.

Quand $H(n-1)$ est satisfaite, chaque chemin a son avant dernier noeud avant n, et donc $d(j)$ est le longueur du plus court chemin de 1 à j.

9.5 PCC avec Distances Non-negatives

Algorithme de DIJKSTRA

1.begin
2. $S = \emptyset, \bar{S} = V$
3. $d(i) \leftarrow \infty \forall i \in V$
4. $d(s) \leftarrow 0$, pred$(s) \leftarrow \emptyset$
5. while $|S| < n$, do
6. begin
7. let $i = \arg\min\{d(j) : j \in \bar{S}\}$
8. $S \leftarrow S \cup \{i\}$
9. $\bar{S} \leftarrow \bar{S} \setminus \{i\}$
10. for each $j \in V^+(i)$, do
11. if $d(j) > d(i) + c_{ij}$, then $d(j) \leftarrow d(i) + c_{ij}$, and pred$(j) \leftarrow i$
12. end
13. end

De nouveau l'opération de la ligne 7 s'appelle: Selection du Noeud, et l'opération de la ligne 11 s'appelle: Mise à jour des Distances.

Theorem 9.8 *L'algorithme de Dijkstra est correct, et de complexité $O(n^2)$.*

Preuve. Soit $D(j)$ le longueur de PCC de s à j. Hypothèse Inductive: A la fin de la boucle (6)-(12), $d(j) = D(j)$ pour tout $j \in S$, et $d(j)$ est la longueur d'un PCC jusquà j dont tous les noeuds sauf le dernier sont dans S (et donc $d(j) \geq D(j)$).

Quand $|S| = 1, S = \{s\}$ et l'hypothèse est vérifiée.

Maintenant soit $|S| = k$ et i le noeud choisi en ligne 7. Nous montrons d'abord que en arrrivant en ligne 12 $d(i) = D(i)$. Supposons le contraire et $d(i) > D(i)$. Alors le chemin optimal de s à i doit avoir son avant dernier noeud dans $\bar{S}$. Soit j le premier noeud de ce chemin qui n'est pas dans S. Le souschemin de s à j doit être optimal. Donc $d(k) = D(k)$. Aussi parce que $c_{ij} \geq 0$, $D(k) \leq D(i)$. Donc $d(k) < d(i)$, ce qui contredit le choix en ligne 7. Conclusion $d(i) = D(i)$.

Pour $j \notin S \cup \{i\}$, $d(j) = \min[d(j), d(i) + c_{ij}]$ en ligne 11. Donc arrivé ensuite en ligne 8, $d(j)$ est le longueur du PCC avec avant dernier noeud soit en S, soit en i, et donc en $S \cup \{i\}$.

9.6 Tours Eulériens et Hamiltoniens

Definition 9.3 Chemin hamiltonien Un chemin (orienté) dans un graphe (orienté) G qui passe par *tous* les noeuds de G et qui a une longueur $|V| - 1$ (c.à.d. qu'il passe par $|V| - 1$ arêtes) est un **chemin hamiltonien**
(Il est sous-entendu mais évident qu'un tel chemin ne passe qu'une fois par chaque noeud.)

Definition 9.4 Cycle hamiltonien Un **cycle hamiltonien** est un cycle[1] qui passe par tous les sommets du graphe (et qui, par définition du cycle, n'y passe qu'une seule fois).

Definition 9.5 Un **graphe hamiltonien** est un graphe qui contient un tour.

Definition 9.6 Un **parcours eulérien** est un parcours qui passe une fois (et seulement une fois) par toutes les arêtes du graphe.

Definition 9.7 Tour eulérien Un **tour eulérien** est un parcours eulérien dont les sommets origine et terminus sont identiques.

Definition 9.8 graphe eulérien Un **graphe eulérien** est un graphe qui contient un tour eulérien.

Rappels

1. Le *degré* d'un noeud est le nombre d'arêtes incidentes à ce noeud dans le graphe.
2. Le nombre de noeuds de degré impair dans un graphe est pair.

Theorem 9.9 *Un graphe connexe possède un parcours Eulérien* **ssi** *le nombre de noeuds de degré impair est 0 ou 2.*

Preuve

- Si le graphe G possède un parcours eulérien, alors tout sommet autre que l'origine ou le terminus du parcours doit être de degré pair.
- Si G est connexe (non trivial) et que le nombre de noeuds de degré impair est 0, alors en partant de n'importe quel sommet il est facile de construire un parcours fermé qui ne passe pas deux fois par la même arête (on part

[1] On dit parfois *tour* pour cycle.

par une arête et on quitte chaque noeud au hasard par une arête non encore utilisée).

Soit C un tel parcours fermé de longueur maximale. Supposons par l'absurde que C n'est pas un tour eulérien.

Soit $G' = G - E(C)$ (*on enlève de G les arêtes dans C*). Par construction tout sommet dans C doit être de degré pair (voir premier •) Donc tout sommet de G' est de degré pair. On peut donc construire dans G' un parcours K fermé ne passant pas deux fois par la même arête. Comme G est connexe, on peut construire K tel que $K \cup C$ soit connexe.

$K \cup C$ est encore un parcours fermé ne passant pas deux fois par la même arête et de longueur $> C$ $\rightarrow$contradiction $\rightarrow$ C est un tour eulérien.

- Soit G connexe et possédant deux sommets, u et v, de degré impair. Soit $G + e$ le graphe obtenu en joignant u à v par une arête e supplémentaire $\rightarrow$ $G+e$ a tous les sommets de degré pair et doit contenir un tour eulérien $\rightarrow$ G doit contenir un parcours eulérien en ôtant e du tour eulérien. ∎

Corollaire
Un graphe connexe possède un tour eulérien **ssi** le nombre de noeuds de degré impair est zéro.

La preuve de ce corollaire est une partie de la preuve du théorème précédent.

Example 9.1 *Ponts de Königsberg (Euler, 1736)*

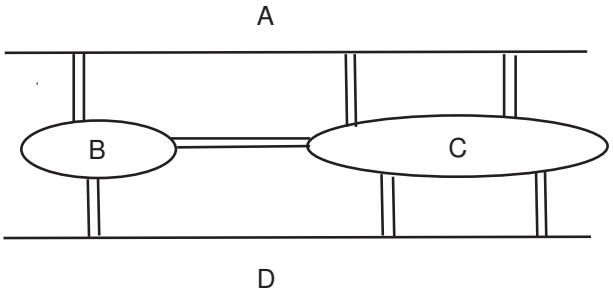


Figure 9.2: Ponts de Königsberg

Examinez la figure 5.1. Est-il possible de traverser chaque pont exactement une fois lors d'une promenade en ville?

La question revient à se poser le problème d'existence d'un parcours eulérien dans le graphe de la Figure 9.2.

Or, le nombre de sommets de degré impair est 4. Donc un tel parcours n'existe pas.

Rappelons-nous l'idée de complexité introduite au Chapitre ?. Supposons que nous voulons une preuve rapide que le graphe est eulérien ou ne l'est pas. C'est facile:

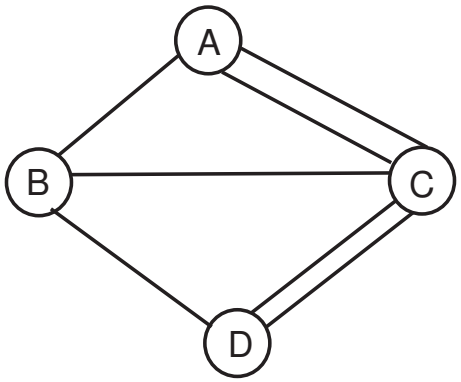


Figure 9.3: Existe-t-il un parcours Eulérien dans ce graphe?

- Pour une réponse "oui", il faut vérifier que $d(v)$ est pair $\forall v \in V$
- Pour une réponse "non", il faut montrer un noeud v avec $d(v)$ impair

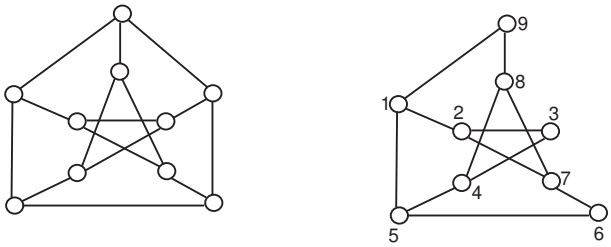


Figure 9.4: Graphe de Petersen

Le problème de caractérisation d'un graphe hamiltonien est beaucoup plus compliqué à résoudre. Par exemple, le graphe de la Figure 9.3 —dit graphe de Petersen— à 10 sommets n'est pas hamiltonien, mais le graphe à 9 sommets obtenu en éliminant n'importe quel sommet est hamiltonien. ¶

Considérons un des sous-graphes à 9 sommets dans la Figure 9.3

Pour vérifier le "oui" il suffit de vérifier que $(e_1, e_2, \ldots, e_9)$ est un tour hamiltonnien.

Mais pour le graphe à 10 sommets, un moyen pour vérifier le "non" n'est pas du tout évident à trouver.

Ceci est une illustration de deux problèmes voisins dont la complexité est fondamentalement différente.

Il peut paraître intéressant d'introduire ici (très brièvement) la théorie de la complexité des algorithmes.

9.7 Théorie de la complexité *(une brève introduction)*

Soit L la longueur des données nécessaires pour décrire un problème (*taille du problème*).

Un problème est dans P si, pour toute instance de longueur L (exemple particulier à ressoudre) le nombre d'opérations de calcul nécessaires à sa résolution est $O(L^p)$ c.à.d. $\leq c_p L^p + c_{p-1} L^{p-1} + \ldots$ pour un certain p fini.

Un problème est dans NP s'il est possible de vérifier en un temps polynomial si une solution donnée satisfait les propriétés cherchées. (On peut aussi présenter la classe NP comme l'ensemble des problèmes solubles en un temps polynomial par des algorithmes non déterministes capables d'envisager un nombre infini de solutions en même temps.)

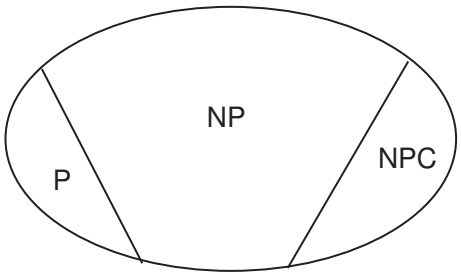


Figure 9.5: $P \stackrel{?}{=} NP$

Un problème est "NP **complet**" s'il fait partie des problèmes les plus compliqués de NP pour lesquels le fait de trouver un algorithme polynomial impliquerait que **toute** la classe NP peut être résolue en un temps polynomial ($P = NP$) (Voir la figure 6.5).

Nous avons vu plusieurs exemples de problèmes dans P:

1. Trouver le pgcd de 2 entiers.
2. Calculer $F(n)$.
3. Trouver la façon optimale de multiplier k matrices.
4. Résoudre: Un graphe $G = (V, E)$ est-il eulérien?
5. Résoudre: Un graphe $G = (V, E)$ est-il non-eulérien?
6. Trouver un arbre de poids maximum.
7. Problème du "plus court chemin" dans un digraphe sans cycle de longueur négative.

Par ce qu'on vient d'observer, le problème: *Un graphe $G = (V, E)$ est-il hamiltonien?* est dans la classe NP, parce qu'on peut rapidement vérifier que la réponse est "oui" si on nous donne un tour.

Presque tous les problèmes d'intérêt pratique pour lesquels on n'a pas trouvé jusqu'à présent un bon algorithme, sont des problèmes NP complets:

- Un graphe est-il hamiltonien?
- Les problèmes d'ordonnancement.
- Les problèmes d'horaire.
- Les problèmes de distribution.
- Les problèmes de localisation.
- Les problèmes de *satisfiabilité* en Logique des propositions.
- etc.

Chapter 10

Encore des Graphes

Aujourd'hui nous allons regarder quelques autres problèmes classiques associés aux graphes, avant de nous tourner vers quelques applications des arbres.

Notation K_n est le graphe complét sur n sommets.

$K_{m,n}$ est le graphe bipartite complét avec m sommets d'un côté et n sommets de l'autre côté.

$G' = (V', E')$ est un sousgraphe de $G = (V, E)$ si $V' \subseteq V$, $E' \subseteq E$, et $\forall\ e = (i, j) \in E', i, j \in V'$.

$G' = (V', E')$ est le graphe induit par V' si $E' = \{e = (i, j) : i, j \in V'\}$.

10.1 Graphes Planaires

Peut-on raccorder trois maisons aux trois utilités, gaz, électricité et eau sans qu'aucune canalisation ne se croise? Voir Figure 10.1.

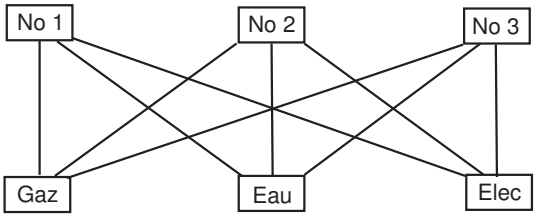


Figure 10.1: Est-ce que $K_{3,3}$ est planaire?

Autrement dit, peut-on dessiner le graphe bipartite complét $K_{3,3}$ sans que les arêtes ne se croisent?

Definition 10.1 Un graphe $G = (V, E)$ est *planaire* s'il peut être tracé dans le plan sans qu'aucun de ses arêtes ne croise un autre. Un tel dessin est appelé un *représentation planaire* de G.

Exemple K_4 est planaire, voir Figure 10.2

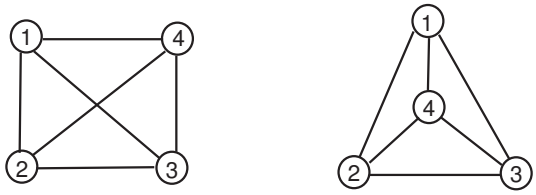


Figure 10.2: K_4 est planaire

Proposition 10.1 *$K_{3,3}$ n'est pas planaire.*

Preuve Soit $V = \{v_1, \ldots, v_6\}$ avec $\{v_1, v_2, v_3\}$ du même côté de la bipartition. $v_1v_4v_2v_5$ forment une courbe fermée C_1 qui divise le plan en deux régions (*faces*) R_1 à l'extérieur et R_2 à l'intérieur.
CAS 1. Supposons que $v_3 \in R_2$. v_3 est lié à v_4 et v_5 coupant R_2 en deux sousrégions R_{21} et R_{22}. Voir Figure 10.3

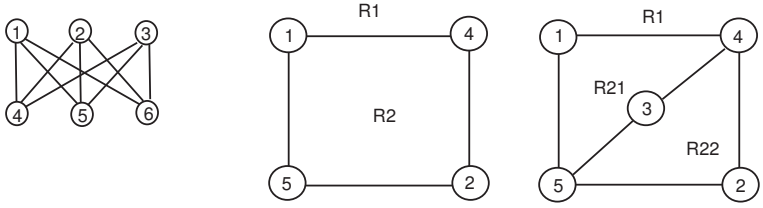


Figure 10.3: Preuve de nonplanarité de $K_{3,3}$

Essayons maintenant de placer v_6. Si $v_6 \in R_1$, on ne peut pas le joindre à v_3 sans couper la courbe C_1.
Si $v_6 \in R_{21}$, on ne peut pas le joindre à v_2 sans couper la frontière de R_{21}.
Si $v_6 \in R_{22}$, on ne peut pas le joindre à v_1 sans couper la frontière de R_{22}.
CAS 2. Si $v_3 \in R_1$, la face extérieure, l'argument est similaire.

Observation K_5 n'est pas planaire.

10.2 Formule d'Euler

Une représentation planaire divise le plan en régions, appelées *faces*.

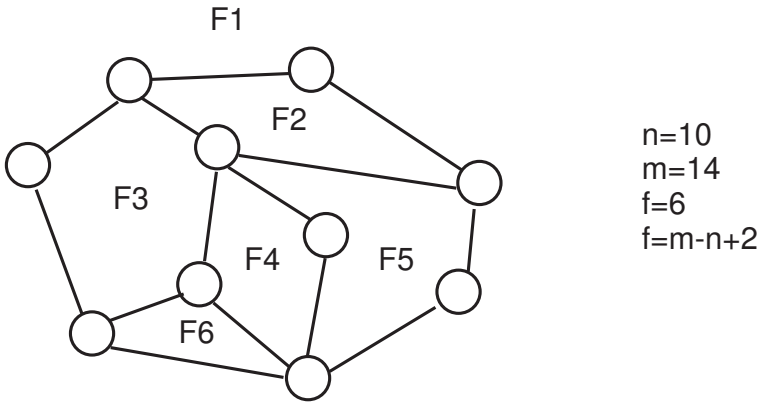


Figure 10.4: Graphe Planaire: Formule d'Euler

Theorem 10.2 Formule d'Euler. *Soit G un graphe planaire connexe avec n sommets, m arêtes et f faces, alors*

$$f = m - n + 2.$$

Preuve Prenons une représentation planaire, et enlevons des arêtes $E' \subseteq E$ pour laisser un arbre $T = (V, E \setminus E')$.
Pour un arbre, $f = 1,\ m = n - 1$, et alors $m - n + 2 = 1 = f$.
Rajoutons les arêtes de E' une par une. Les deux extrémités de e forment partie d'une face F. En ajoutant cet arête, on augmente le nombre d'arêtes et le nombre de faces par 1, et le reste suit par induction - on espère. Voir Figure 10.5.

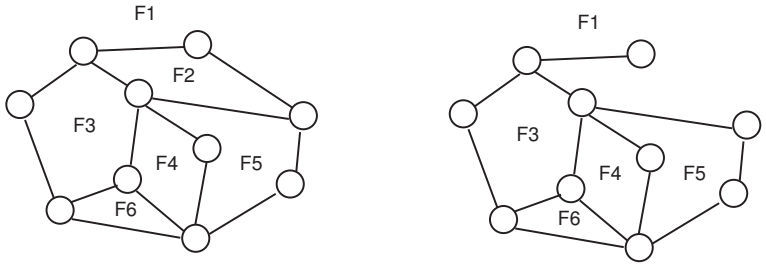


Figure 10.5: Preuve de Formule d'Euler

Proposition 10.3 *Dans un graphe simple planaire connexe avec $n \geq 3$,*

$$m \leq 3n - 6.$$

Preuve. $m \geq 3f/2$. Donc $m - n + 2 = f \leq 2m/3$.
Corollaire K_5 n'est pas planaire.

Proposition 10.4 *Dans un graphe simple planaire connexe avec $n \geq 3$, et aucun cycle de longueur 3,*

$$m \leq 2n - 4.$$

Corollaire $K_{3,3}$ n'est pas planaire.

Definition 10.2 Soit G un graphe planaire. Un graphe G' obtenu en enlevant un arête (u, v), et en ajoutant un sommet w et deux arêtes (u, w) et (w, v) est clairement planaire.
Cette opération est appelée une *sousdivision élémentaire*. Voir Figure 10.6
Les graphes G_1 et G_2 sont *homéomorphes* s'ils peuvent être obtenus à partir du même graphe par des sousdivisions élémentaires.

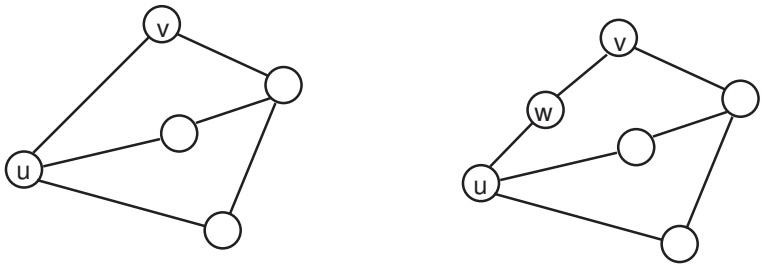


Figure 10.6: Sousdivision Elémentaire

Theorem 10.5 Théorême de Kuratowski. *G est nonplanaire si et seulement s'il contient un sousgraphe homéomorphe à $K_{3,3}$ ou K_5.*

Exemple Le graphe de Petersen n'est pas planaire. Il contient un homéomorphe de $K_{3,3}$. Prendre $V_1 = \{1, 4, 8\}, V_2 = \{3, 5, 6\}$.

10.3 Coloriage des Graphes

Definition 10.3 Soit G un graphe simple.
Une fonction $f : V \to \{1, \ldots, k\}$ est un k-coloriage de G si $f(i) \neq f(j)$ pour tout $(i, j) \in E$.
Le nombre chromatique de G est le plus petit k tel que G a un k-coloriage.

Theorem 10.6 Théorême des Quatres Couleurs. *Le nombre chromatique d'un graphe planaire ne dépasse pas 4.*

Exemple Quel est le nombre chromatique d'un arbre?
Quel est le nombre chromatique d'un graphe bipartite?
Quel est le nombre chromatique d'un cycle de longueur impair (sans chordes)?
Quel est le nombre chromatique d'un clique (sousgraphe complét)?

Application à des Horaires
Soit $i \in V$ l'ensemble des examens de l'IAG, et E une liste des pairs de cours qui partagent au moins un(e) étudiant(e). Si on insiste pour qu'aucun étudiant doit passer deux examens le même jour, le nombre minimum de jours nécessaires pour compléter la session est donné par le nombre chromatique de G.

10.4 Encore des Arbres

Etant donné un arbre $T = (V, E)$, si on choisit un sommet particulier comme *racine*, on peut attribuer une direction à chaque arc.

Definition 10.4 Un arbre avec racine r où chaque arc est orienté (i, j) si i est plus proche de la racine que j est *une arborescence*.

Definition 10.5 Si T est une arborescence, et $v \neq r$, le *père* de v est le sommet unique u tel que $(u, v) \in E$. u est un *fils* de v si $(v, u) \in E$.
Les *ancêtres* de v sont les sommets sur le chemin entre r et v.
Les *descendants* de v sont tous les sommets $u \neq v$ tels que le chemin de u à la racine passe par v.
Un sommet sans fils est une *feuille* de l'arborescence (ou de l'arbre).
Les sommets avec fils sont les *sommets internes*
Une sous-arborescence avec racine a est l'arbre constitué de a, ses descendants et tous les arcs inhérents à ces descendants.

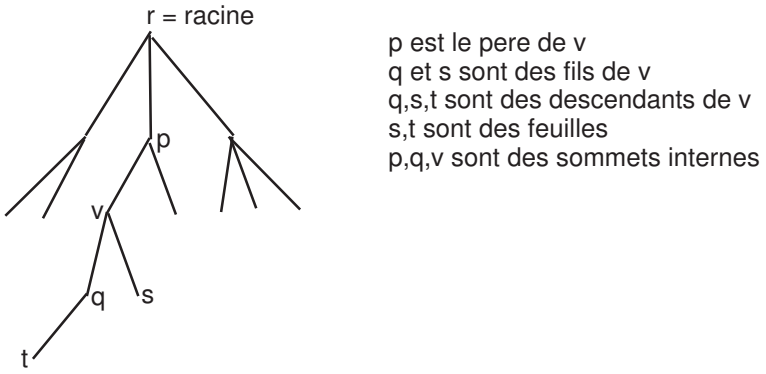


Figure 10.7: Une arborescence

Definition 10.6 Une arborescence est un *arbre k-aire* si chaque sommet interne a un maximum de k fils. Il est un arbre k-aire *complet* si chaque noeud interne a exactement m fils. Si $k = 2$, on parle d'un *arbre binaire*.

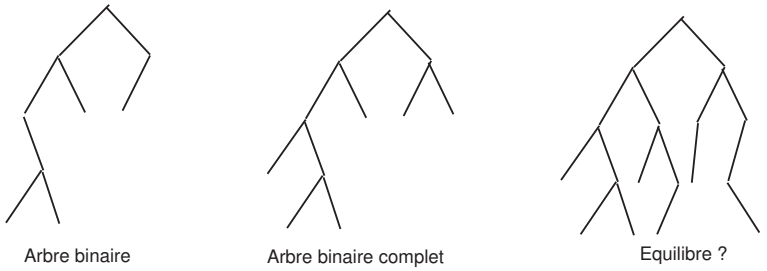


Figure 10.8: Arbres binaires

On parle d'un arbre *ordonné*, si les fils sont ordonnés de gauche à droite.

10.4.1 Propriété des arbres

Observation 10.1 *Soit i le nombre de noeuds internes et l le nombre de feuilles.*
Un arbre k-aire complet avec

i n sommets comporte $i = \frac{n-1}{k}$ sommets internes et $\ell = n - \frac{n-1}{k}$ feuilles.

ii i sommets internes comporte $n = mi + 1$ sommets et $(m-1)i + 1$ feuilles.
Il suffit de voir que $i + l = n$ et $n = ki + 1$.

Definition 10.7 La *hauteur* ou *profondeur* d'une arborescence est le nombre maximum de sommets sur une chaine de la racine à une feuille.
Une arborescence m-aire de hauteur h est *équilibrée* si toutes les feuilles sont au niveau h ou $h - 1$ (le niveau de la racine = 1).

Theorem 10.7 *Le nombre maximum de feuilles dans un arbre k-aire de hauteur h est k^h.*
Le nombre maximum de noeuds $= \frac{k^{n+1} - 1}{k - 1}$.

Corollary 10.1 *Dans un arbre k-aire de hauteur h avec ℓ feuilles*

$$h \geq \lceil \log_k \ell \rceil$$

10.5 Arbre de Feuille Binaire (Retour)

Avant nous avons cherché un élément dans une liste ordonnée

1 3 4 5 6 13 27 43 56 82.

L'algorithme du Chapitre 6 travaillait sur l'hypothèse qu'on pouvait trouver l'élément au milieu de la liste en $O(1)$.
Ici on propose de garder les éléments dans un arbre binaire. Nous partons de la liste non ordonnée

6 56 13 4 3 27 82 43 1 5

Le premier élément devient la clef de la racine. Ensuite pour ajouter un nouvel élément, on compare avec la racine et on se *déplace à gauche* si l'élément est plus petit que la clef du sommet respectif, et le sommet a un fils de gauche. S'il n'y a plus de fils de gauche, alors un nouveau sommet est créé ayant cet élément comme clef. Sinon on se déplace à droite, etc.

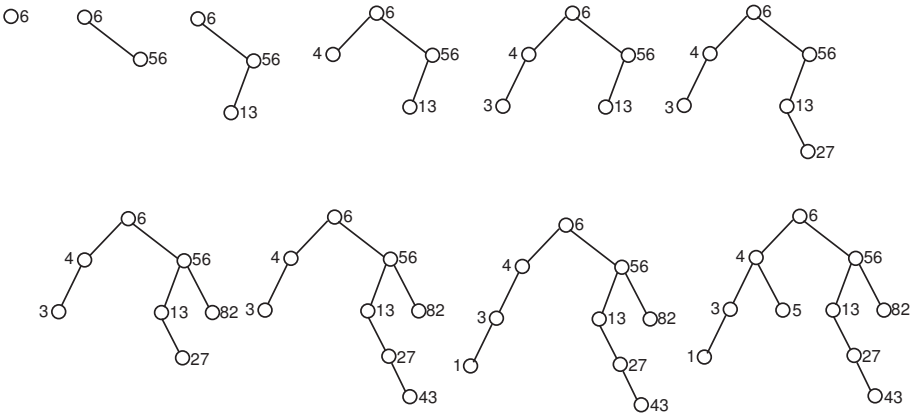


Figure 10.9: Fouille binaire

Le test pour voir si $x = 42$ s'y trouve est le même que pour la construction de l'arbre.
On compare 42 avec 6.
Parce que $42 > 6$, on descend vers la droite.
Parce que $42 < 56$, on descend à gauche.
Parce que $42 > 13$, on descend à droite.
Parce que $42 > 27$, on descend à droite.
Parce que $42 < 43$, un veut descendre à gauche! Il n'y a pas de fils, donc 42 n'est pas dans la liste.

Observation. Il faut au plus h comparaisons. Si l'arborescence est équilibrée, $h \leq \lceil \log_2 n \rceil$. Donc pour être efficace, il faut eąuilibrer l'arbre, et le rééquilibrer après chaque modification!

10.6 Parcours d'un arbre

Les arborescences ordonnées sont souvent utilisées pour stocker des données et pour représenter différents types d'expressions.

10.6.1 Systèmes d'Adressage Universel

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

2. Pour chaque sommet v au niveau k avec l'étiquette A, on étiquette ses k_v fils, comme ils sont dessinés de gauche à droite, avec $A.1, \cdots, A.k$.

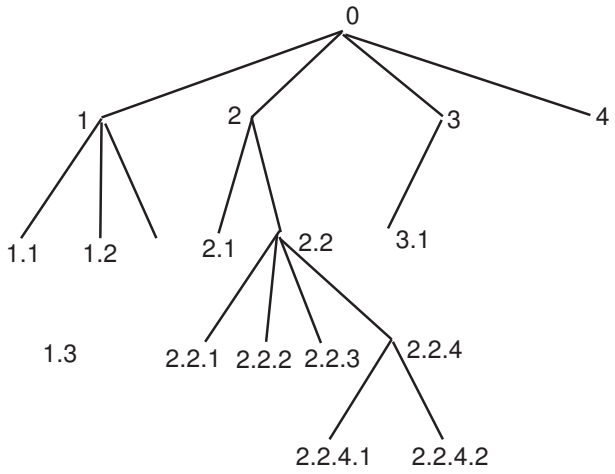


Figure 10.10: Adressage Universel

Definition 10.8 *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 $T_1, \cdots, T_n$ sont les sousarbres de r de gauche à droite dans T.
Le *parcours préfixe* débute en parcourant r, et ensuite le parcours préfixe de T_1, après de T_2, etc.

Le parcours préfixe de l'arborescence de la Figure 10.10 donne

0	1	1.1	1.2	1.3	2	2.1	2.2	2.2.1
2.2.2	2.2.3	2.2.4	2.2.4.1	2.2.4.2	3	3.1	4	

Notons que c'est l'ordre lexicographique.

Definition 10.9 L'Ordre Lexicographique

$$a_1 \dots a_m \prec_L b_1 \dots b_n$$

s'il existe $i \leq \min[m, n]$ tel que
i) $a_j = b_j$ pour $j = 1, \dots, i-1$, et $a_i < b_i$, ou
ii) $m < n$ et $a_j = b_j$ pour $j = 1, \dots, m$.

Donc 4237 $\prec_L$ 424 et 4237 $\prec_L$ 4237162.

Definition 10.10 *Ordre infixe.*
Si T est uniquement r, r est le *parcours infixe* de T. Sinon le *parcours infixe* débute avec le parcours infixe de T_1, après avec r, et après le parcours infixe de $T_2, \cdots$, après T_n.

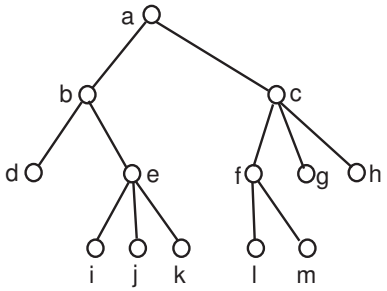


Figure 10.11: Un arborescence

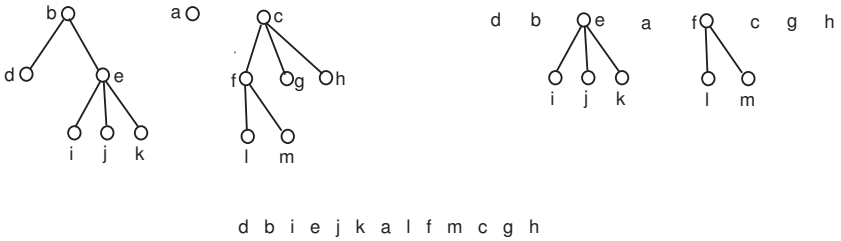


Figure 10.12: Ordre Infixe

Example 10.1 *L'ordre préfixe est*

$$a\ b\ d\ e\ i\ j\ k\ c\ f\ l\ m\ g\ h.$$

L'ordre infixe pour l'arborescence de la Figure 10.11 est

$$d\ b\ i\ e\ j\ j\ k\ a\ \ell\ m\ c\ g\ h.$$

Une façon de l'obtenir est montrée dans la Figure 10.12

10.6.2 Représentations des Expressions

On peut aussi utiliser les arbres binaires pour représenter les expressions. Les opérateurs sont associés aux sommets internes et les variables ou constantes aux feuilles.

Example 10.2 *Une arborescence ordonnée binaire qui représente l'expression*

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

est montrée à la Figure 10.2.

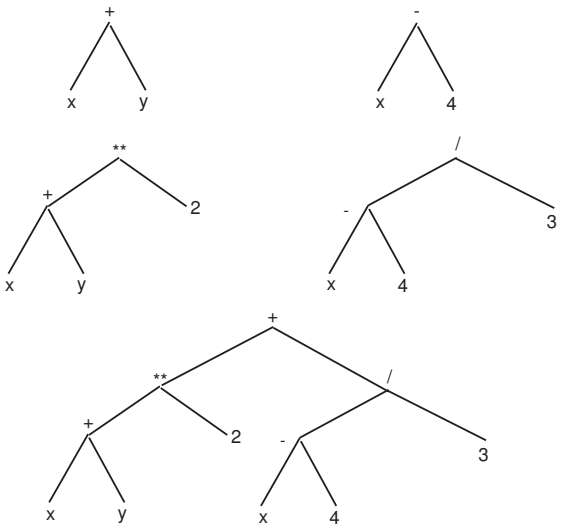


Figure 10.13: Expression

La forme infixe de cette arborescence est

$$((x + y) \uparrow 2) + ((x - 4)/3).$$

La forme préfixe de cette arborescence est

$$+ \ \uparrow \ + \ x \ y \ 2 \ / \ - \ x \ 4 \ 3.$$

N.B. Pour rendre les expressions infixes claires il faut des parenthèses. Le prochain exemple montre cette ambiguité.

Example 10.3 $x+y/x+3$ *est l'expression infixe de* $(x+y)/(x+3)$, $(x+(y/x))+3$ *et* $x + (y/(x + 3))$. *Voir la Figure 10.14.*

Avec la forme préfixe ou postfixe (notation ...), il n'y a pas d'ambiguité.

Le dernier exemple montre comment évaluer une expression préfixe.

Example 10.4 *Quelle est la valeur de l'expression préfixe*

$$+ - \star 235/ \uparrow 234$$

On part de droite à gauche, et on cherche le premier opérant

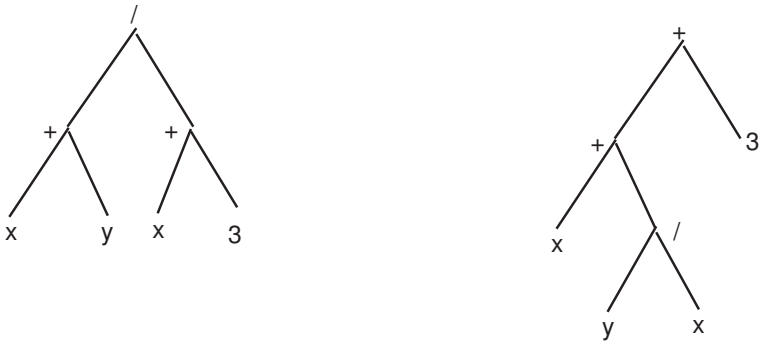


Figure 10.14: Ambiguité?

$$
\begin{array}{cc}
\uparrow 23 = 8 & 2 \uparrow 3 = 8 \\
+ - \star 235/84 & \\
/84 = 2 & 8/4 = 2 \\
+ - \star 2352 & \\
\star 23 = 6 & 2 \times 3 = 6 \\
+ - 652 & \\
-65 = 1 & 6 - 5 = 1 \\
+12 = 3 & 1 + 2 = 3.
\end{array}
$$

Chapter 11

Ordres Partiels et Treillis

Nous avons déjà vu des relations particulières, par exemple: une relation d'équivalence qui est réflexive, symétrique, transitive et

$$aRa \qquad aRb \Rightarrow bRa, \qquad aRb \text{ et } bRc \Rightarrow aRc.$$

Ici nous étudions une autre relation importante.

11.1 Ordres Partiels

Definition 11.1 Une relation R définie sur un ensemble A est une **relation de préordre**

si R est	réflexive	aRa
"	antisymétrique	aRb et $bRa \Rightarrow a = b$
"	transitive	aRb et $bRc \Rightarrow aRc$.

On écrit $\preceq$ à la place de R.

Definition 11.2 Un **Ensemble partiellement ordonné** ou un **ordre partiel** est un ensemble A muni d'une relation de préordre, qu'on écrit $(A, \preceq)$.

Example 11.1 *La relation de divisibilité $|$ est une relation de préordre dans l'ensemble des entiers positifs puisqu'elle est réflexive, antisymétrique et transitive. On voit que $(Z^+, |)$ est un ensemble partiellement ordonné. (Z^+ signifie l'ensemble des entiers positifs.)*

Example 11.2 *Démontrez que la relation d'inclusion $\subseteq$ constitue une relation de préordre dans l'ensembe des parties d'un ensemble S.*
Puisque $A \subseteq A$ lorsque A est un sous-ensemble de S, $\subseteq$ est réflexive. Elle est antisymétrique puisque $A \subseteq B$ et $B \subseteq A$ impliquent que $A = B$. Finalement, $\subseteq$ est transitive, puisque $A \subseteq B$ et $B \subseteq C$ impliquent que $A \subseteq C$. Ainsi, $\subseteq$ est une relation de préordre dans $\mathcal{P}(\mathcal{S})$, et $(\mathcal{P}(\mathcal{S}), \subseteq)$ est un ensemble partiellement ordonné.

Definition 11.3 Etant donné un ordre partiel $(A, \preceq)$, deux éléments a et b sont **comparables** si $a \preceq b$ ou $b \preceq a$.

Definition 11.4 Etant donné un ordre partiel $(A, \preceq)$, deux éléments a et b sont **incomparables** si a n'est pas en relation avec b c.à.d. ni $a \preceq b$ ni $b \preceq a$.

Example 11.3 *Dans l'ensemble partiellement ordonné $(Z^+, |)$, les entiers 3 et 9 sont-ils comparables ? Les entiers 5 et 7 sont-ils comparables ?*
Les entiers 5 et 7 sont incomparables, car $5 \not| 7$ et $7 \not| 5$. Les entiers 3 et 9 sont comparables puisque $3 \mid 9$.

Definition 11.5 Un ordre partiel $(A, \preceq)$ est un **ordre total** si toute paire d'éléments de A est comparable.

Example 11.4 *L'ensemble partiellement ordonné $(Z, \leq)$ est totalement ordonné puisque $a \leq b$ ou $b \leq a$ lorsque a et b sont des entiers.*

Example 11.5 *L'ensemble partiellement ordonné $(Z^+, |)$, n'est pas totalement ordonné puisqu'il contient des éléments qui sont incomparables, tels que 5 et 7.*

N.B. Si $x \preceq y$ on peut écrire $y \succeq x$;
et si $x \preceq y$ et $x \neq y$, on peut écrire $x \prec y$.

11.2 Ordre de classement

Les mots d'un dictionnaire sont présentés par ordre alphabétique. Il s'agit d'un cas particulier d'ordre des chaînes dans un ensemble élaboré à partir de l'ordre partiel de l'ensemble. Cette construction fonctionne dans un ensemble partiellement ordonné de la manière décrite ci-après.

On montre d'abord comment former une relation de préordre du produit cartésien de deux ensembles partiellement ordonnés, $(A_1, \preceq_1)$ et $(A_2, \preceq_2)$. L'**ordre de classement** $\preceq$ de $A_1 \times A_2$ est défini en précisant qu'un couple est plus petit qu'un autre si le premier élément du premier couple est plus petit (dans A_1) que le premier élément du deuxième couple, ou si les premiers éléments sont égaux, mais que le deuxième élément de du premier couple est plus petit (dans A_2) que le deuxième élément du deuxième couple. En bref, (a_1, a_2) est plus petit que (b_1, b_2) c'est-à-dire

$$(a_1, a_2) \prec (b_1, b_2),$$

soit si $a_1 \prec_1 b_1$, soit si $a_1 = b_1$ et $a_2 \prec_2 b_2$.

Example 11.6 *A noter que $(1, 2, 3, 5) \prec (1, 2, 4, 3)$, puisque les éléments dans les deux premières positions de ces 4-uples concordent mais que, dans la troisème position, l'élément dans le premier 4-tuple, soit 3, est inférieur à celui dans le deuxième 4-tuple, soit 4. (Ainsi, l'ordre des 4-tuples est l'ordre lexicographique qui provient de la relation habituelle "plus petit que ou égal à" dans l'ensemble des entiers.)*

Example 11.7 *Considérez l'ensemble des chaînes des lettres miniscules françaises. Utilisez l'ordre des lettres de l'alphabet pour former un ordre de classement dans l'ensemble des chaînes. Une chaîne est inférieure à la deuxième chaîne si la lettre de la première chaîne dans le première position où les chaînes diffèrent est placée avant la lettre dans la deuxième chaîne dans cette même position, ou si la première chaîne et la deuxième chaîne concordent dans toutes les positions, mais que la deuxième chaîne a plus de lettres. Cet ordre est le même que celui qui est utilisé dans les dictionnaires. Par exemple,*

$$discreet \prec discrete,$$

puisque ces chaînes diffèrent d'abord dans la 7^e position et que $e \prec t$. De plus,

$$discreet \prec discreetness,$$

puisque les huit premières lettres concordent, mais que la deuxième chaîne est plus longue. De plus,

$$discreet \prec discretion,$$

puisque

$$discreet \prec discreti.$$

11.3 Diagramme de Hasse

Definition 11.6 Les ordres partiels peuvent se représenter graphiquement par ce qu'on appelle un **diagramme de Hasse**:

- les éléments sont représentés par des sommets;
- a et b sont joints par une arête **ssi**
  $a \preceq b$ et $\nexists z (\neq a, b)$ t.q. $a \preceq z$ et $z \preceq b$.

Example 11.8 *Considérons l'ordre partiel:*
$(\{1, 2, 3, 4, 6, 8, 12\}$*, Division entière)*
$x \preceq y \iff x$ *divise* y.
Son diagramme de Hasse est montré dans la Figure 11.1.

Example 11.9 *Considérons l'ordre partiel:*
$\big(\mathcal{P}(\{1, 2, 3\}), \subseteq\big)$.
Son diagramme de Hasse est montré dans la Figure 11.2.

11.3.1 Elément maximal, maximum, ...

Soit $(A, \preceq)$ un ordre partiel:

Definition 11.7 Un élément $x \in A$ est **maximal** si $\nexists y \in A \ : \ x \prec y$.

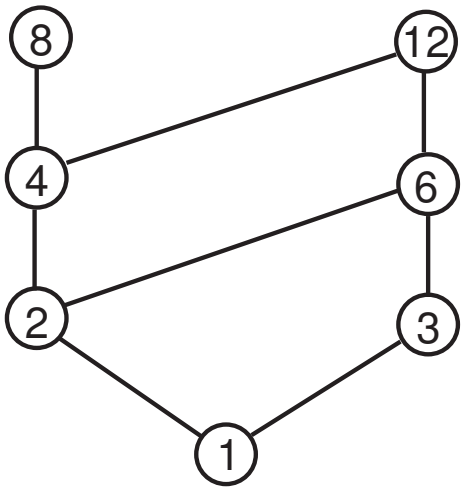


Figure 11.1: Diagramme de Hasse de $(\{1, 2, 3, 4, 6, 8, 12\}, |)$

Definition 11.8 Un élément $x \in A$ est **minimal** si $\nexists y \in A \ : \ y \prec x$.

Definition 11.9 Un élément $x \in A$ est **le maximum** (le plus grand, noté "1") si $y \preceq x \quad \forall y \in A$.

Definition 11.10 Un élément $x \in A$ est **le minimum** si $x \preceq y \quad \forall y \in A$ (c.à.d. le plus petit; noté "0").

Example 11.10 *Dans l'exemple de la Figure 11.1 :*

- **8** *et* **12** *sont maximaux;*
- $\nexists$ *maximum;*
- **1** *est à la fois minimal et minimum.*

Un ordre partiel fini possède nécessairement des éléments minimaux et maximaux. Si le maximum(minimum) existe, il est unique.

Definition 11.11 Soit $(A, \preceq)$ un ordre partiel et $S \subseteq A$. $z \in A$ est une **borne supérieure/ majorant** de $S \subseteq A$ si $x \preceq z \ \forall x \in S$.

Definition 11.12 Soit $(A, \preceq)$ un ordre partiel et $S \subseteq A$. $z \in A$ est le *supremum* de S si z est un majorant de S, et z est un minorant des majorants de S. Si $x, y \in Z$ ont un supremum, il est écrit $x \vee y$, et un *infimum* est écrit $x \wedge y$.

Example 11.11 *Dans l'exemple de divisibilité des entiers de l'exemple 1 avec* $(\{1\ 2\ 3\ 4\ 6\ 8\ 12\}, |)$.

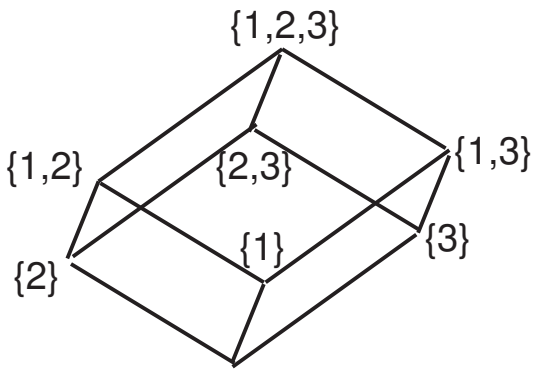


Figure 11.2: Diagramme de Hasse de $(\mathcal{P}\ \{1,2,3\}, \subseteq)$

- $2 \vee 3 = 6$*;*
- $4 \vee 3 = 12$*;*
- *8 et 12 n'ont pas de "supremum".*

Example 11.12 *Y a-t-il un plus grand élément et un plus petit élément dans l'ensemble partiellement ordonné* $(Z^+, |)$ *?*
L'entier 1 est le plus petit élément puisque $1 \mid n$ *lorsque* n *est un entier positif. Puisqu'il n'y a aucun entier divisible pour tous les entiers positifs, il n'y a pas de plus grand élément.*

Proposition 11.1 *Soit* $P = (A, \preceq)$ *un ordre partiel fini. Toute paire d'éléments possède un majorant* $\Longleftrightarrow \exists$ *un élément maximum dans* A*.*

Preuve

- Si un maximum z existe, alors $\forall x, y \in A$, z est une borne supérieure et x, y possèdent un majorant.
- s'il n'existe pas de maximum, alors il existe deux éléments qui n'ont pas de borne supérieure. ∎

On constate donc que la question intéressante concerne l'existence du supremum.

Example 11.13 *Dans la Figure 11.3*
a et b sont tous deux majorants de c, d *(donc* $c \vee d$ *n'est pas défini).*

Example 11.14 *Partitions d'un entier*
Dans la Figure 11.4 on peut constater que chaque paire d'éléments a un majorant, mais toute paire n'a pas un supremum.

En raisonant de façon **duale**, on peut définir:

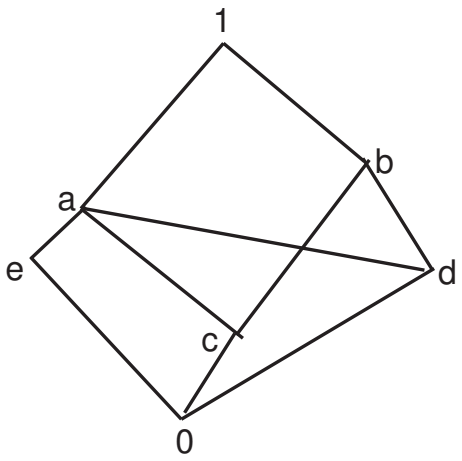


Figure 11.3: Diagramme de Hasse No 3.

borne supérieure	$\leftrightarrow$	borne inférieure
supremum	$\leftrightarrow$	infimum[1]
$\vee$	$\leftrightarrow$	$\wedge$
$\preceq$	$\leftrightarrow$	$\succeq$

Example 11.15
z est une **borne inférieure** *de S si $z \preceq x, \forall\ x \in S$.*

Example 11.16
z est un **infimum** *de S si z est un minorant de S, et z est un majorant des minorants de S. L'infimum de x, y s'écrit $x \wedge y$.*

11.4 Treillis

Definition 11.13 Un *treillis* (LATTICE) est un ordre partiel où chaque paire d'éléments x, y possède un supremum $x \vee y$ et un infimum $x \wedge y$.
On le note

$$L = (A, \vee, \wedge)$$

Dans ce cas, $\vee$ et $\wedge$ sont des opérations binaires.

Example 11.17 *Déterminez si les ensembles partiellement ordonnés représentés par chacun des diagrammes de Hasse de la Figure 11.5 sont des treillis.*
Les ensembles partiellement ordonnés représentés par les diagrammes de Hasse

[1] La plus grande borne inférieure. Voir définition 13.

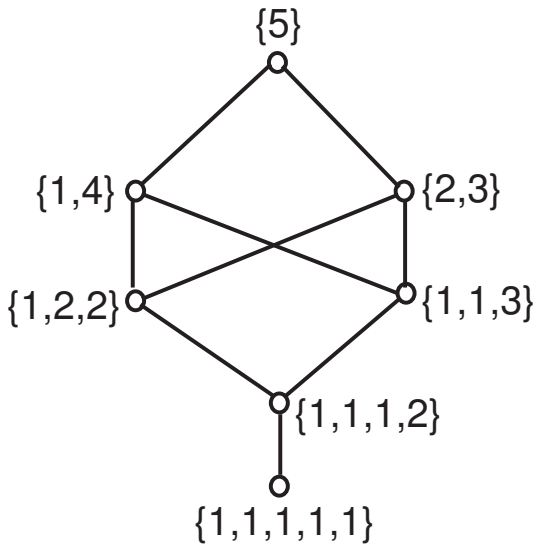


Figure 11.4: Diagramme de Hasse des partitions de 5

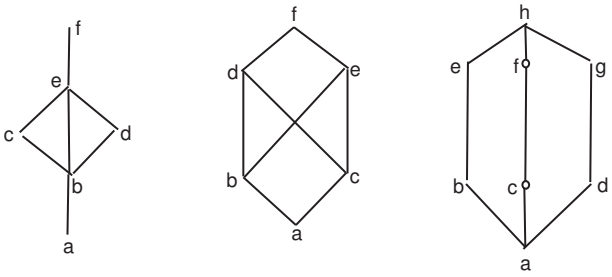


Figure 11.5: Treillis?

en a) et en c) sont tous les deux des treillis puisque, dans chaque ensemble partiellement ordonné, chaque paire d'éléments a un supremum et un infimum, comme le lecteur devra le vérifier. Par ailleurs, l'ensemble partiellement ordoné du diagramme de Hasse présenté en b) n'est pas un treillis, car les éléments b et c n'ont aucun supremum. Pour le vérifier, on note que chacun des éléments d, e et f est un majorant, mais qu'aucun de ces trois éléments ne précède les deux autres par rapport à l'ordre de cet ensemble partiellement ordonné.

Proposition 11.2 *Les relations suivantes sont satisfaites dans tout treillis.*

(i)	$x \vee x = x$	*Idempotence*	$x \wedge x = x$
(ii)	$x \vee y = y \vee x$	*Commutativité*	$x \wedge y = y \wedge x$
(iii)	$(a \vee b) \vee c = a \vee (b \vee c)$	*Associativité*	$(a \wedge b) \wedge c = a \wedge (b \wedge c)$
(iv)	$a \vee (a \wedge b) = a$	*Absorption*	$a \wedge (a \vee b) = a$
(v)	$a \preceq b \Leftrightarrow a \vee b = b$	*Consistance*	$a \succeq b \Leftrightarrow a \wedge b = b$
		DUALITÉ	

Preuve

- Par la dualité, on ne doit démontrer qu'une des colonnes.
- Pour (i), (ii) et (iii) la relation est évidente.
- Pour (iv):

  Soit $z = a \vee (a \wedge b)$ (z existe et est unique dans un treillis)
  Comme $a \succeq a$ (réflexif) et $a \succeq a \wedge b$ (définition de $\wedge$)
  $\Rightarrow a$ est une borne supérieure de a et $a \wedge b$. Donc, par la définition de supremum, $a \succeq z$.

  Mais, $z \succeq a$ par définition de $\vee$.

  Maintenant, par l'antisymétrie de $\preceq$, $z = a$.

- Pour (v):

  Soit $z = a \vee b \Rightarrow z \succeq a$; $z \succeq b$ (définition de $\vee$).
  $a \preceq b \Rightarrow b \succeq a$ et $b \succeq b \quad \Rightarrow b$ est une borne supérieure de a et b.

  Donc, $b \succeq z$ par la définition de supremum,
  $\Rightarrow z = b$

  Inversément $a \vee b = b \Rightarrow b \succeq b$ et $b \succeq a$ par définition de $\vee$ ∎

Example 11.18
Nous avons un treillis avec les données suivantes (voir Figure 11.6)
$\big(\{1, 2, 3, 4, 5, 6, 10, 12, 15, 20, 30, 60\}$, *Division entière*$\big)$
$x \preceq y \Longleftrightarrow x$ *divise* y.

Ceci est un treillis où :
$\vee$ représente le **ppcm**: plus petit commun multiple;
$\wedge$ représente le **pgcd**: plus grand commun diviseur. ∎

Considérons maintenant l'idée d'un sous-ensemble d'un l'ordre partiel ou treillis.
Soit $P = (A, \preceq)$ un l'ordre partiel.
Alors $\forall A' \subseteq A, \quad P' = (A', \preceq)$ est un ordre partiel
(*tout sous-ensemble d'un ordre partiel est un ordre partiel*).

Definition 11.14 Si $L_1 = (A_1, \vee_1, \wedge_1)$ et $L_2 = (A_2, \vee_2, \wedge_2)$ sont des treillis, $L_1 \times L_2 = (A_1 \times A_2, \vee', \wedge')$ est un **treillis produit** où: $(a_1, a_2) \vee' (b_1, b_2) = (a_1 \vee_1 b_1, a_2 \vee_2 b_2)$ et $(a_1, a_2) \wedge' (b_1, b_2) = (a_1 \wedge_1 b_1, a_2 \wedge_2 b_2)$

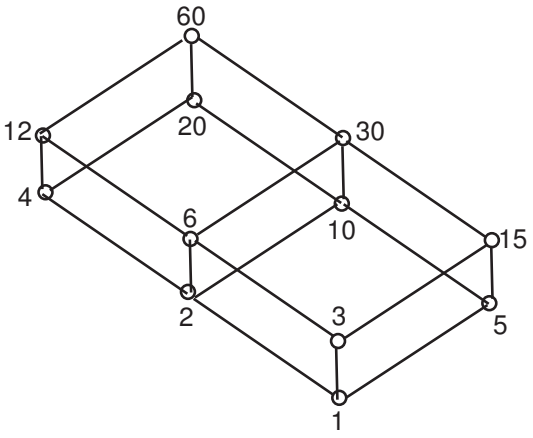


Figure 11.6: Treillis de Diviseurs de 60

Definition 11.15 Un treillis $L' = (A', +, \bullet)$ est un **sous-treillis** de $L = (A, \vee, \wedge)$ si $A' \subseteq A$ et $+, \bullet$ sont les restrictions de $\vee, \wedge$ à A'. C'est-à-dire que le supremum et l'infimum des éléments de A' doivent appartenir à A'.

Example 11.19 Modèle de treillis du flux d'information
Dans plusieurs situations, le flux d'information entre une personne et une autre ou un logiciel et un autre est restreint par un contrôle de sécurité. On peut utiliser un modèle de treillis pour représenter différents règlements relatifs au flux d'information. Par exemple, un règlement courant qui est relatif aux flux d'information est le règlement de sécurité multiniveaux utilisé dans les systèmes gouvernementaux et militaires. On attribue une classe de sécurité à chaque information, et chaque classe de sécurité est représentée par une paire (A, C), où A est un niveau d'autorité et C une catégorie. Les personnes et les logiciels peuvent alors accéder à l'information à partir d'un ensemble de classes de sécurité précis et restreint.

Les niveaux d'autorité conventionnels qu'utilise le gouvernement des Etats-Unis sont les suivants: non secret (0), confidentiel (1), secret (2) et ultra secret (3). Les catégories utilisées dans les classes de sécurité sont les sous-ensembles de l'ensemble de tous les compartiments pertinents à un domaine d'intérêt particulier. Chaque compartiment représente un sujet particulier. Par exemple, si l'ensemble des compartiments est { espions, taupes, agents doubles }, alors il y a huit catégories différentes, une pour chacun des huit sous-ensembles de l'ensemble des compartiments, par exemple {espions,taupes}.

On peut ordonner les classes de sécurité en précisant que $(A_1, C_1) \preceq (A_2, C_2)$ si et seulement si $A_1 \leq A_2$ et $C_1 \subseteq C_2$. L'information peut circuler entre la classe de sécurité (A_1, C_1) et la classe de sécurité (A_2, C_2) si et seulement si $(A_1, C_1) \preceq (A_2, C_2)$. Par exemple, l'information peut circuler entre la classe de sécurité (secret, {espions,taupes}) et la classe de sécurité (ultra secret, {espions,

taupes, agents doubles}), alors que les informations ne peuvent circuler entre la classe de sécurité (ultra secret, {espions,taupes}) et les classes de sécurité (secret, {espions, taupes, agents doubles}) ou (ultra secret, {espions}).

Le lecteur devra démontrer que l'ensemble de toutes les classes de sécurité avec l'ordre défini dans cet exemple forme un treillis.

Example 11.20 *Observons que le treillis des diviseurs L des diviseurs de 60, voir Figure 11.6, est un treillis produit de trois treillis*

$$L_1 = (\{1, 2, 4\}.|), L_2 = (\{1, 3\}.|), L_3 = (\{1, 5\}.|).$$

$L' = (\{1, 2, 4, 5, 10, 20\}, |)$ *est un soustreillis de L,*
lié au fait que $60 = 2^2 \times 3^1 \times 5^1$.

11.5 Tri topologique de nouveau

Etant donné un ordre partiel $(S, \preceq)$, le but ici est de trouver un ordre total $(S, \leq')$ compatible, c'est-à-dire

$$\forall\ x, y \in S,\ \text{si } x \prec y,\ \text{alors } x <' y.$$

Example 11.21 *On a un projet qui consiste en 20 tâches dont certaines doivent en précéder d'autres, mais on ne peut faire qu'une des tâches à la fois, on veut trouver un ordre pour les rendre compatibles avec les contraintes de précédence.*

Observation 1. Tout ordre partiel fini et nonvide $(S, \preceq)$ admet un élément minimal.
Observation 2. Si $(S, \preceq)$ est un ordre partiel et $a \in S$, $(S \setminus a, \preceq')$ est un ordre partiel où $\forall x, y \in S \setminus a, x \preceq' y$ ssi $x \preceq y$.

L'algorithme pour faire un tri topologique suit immédiatement.

$$S^1 = S \qquad t = 1 \qquad n = |S|.$$

Pour $t = 1, \cdots, n$

1. Trouver un élément minimal a^t de S^t
2. $S^{t+1} = S^t / a^t$

A la fin $a_1 <' a_2 <' \cdots <' a_n$ est un ordre total compatible avec $(S, \preceq)$.

Example 11.22 *Un projet de développement dans une entreprise informatique exige l'accomplissement de sept tâches. On ne peut effectuer certaines de ces tâches qu'après que d'autres tâches sont terminées. On définit une relation de préodre pour les tâches en considérant la tâche X $\prec$ la tâche Y, si on ne*

peut commencer la tâche Y avant d'avoir achevé la tâche X. Le diagramme de Hasse pour les sept tâches, par rapport à la relation de préordre, est présenté à la Figure 11.7 Trouver un ordre dans lequel on peut effectuer ces tâches pour exécuter le projet.

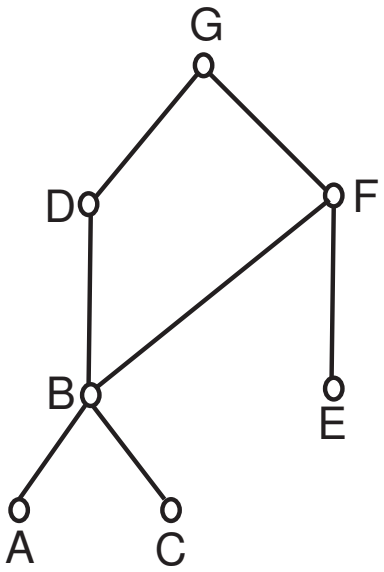


Figure 11.7: Préordre associé aux tâches

On peut obtenir un ordre d'exécution des sept tâches en effectuant un tri topologique. Le résultat de ce tri, soit $A \prec C \prec B \prec E \prec F \prec D \prec G$, représente une possibilité d'ordre d'exécution des tâches.

Chapter 12

Monoïdes, Isomorphismes et Homomorphismes

Après avoir présenté les relations binaires et les fonctions de $A \rightarrow B$ ainsi que les relations sur un ensemble (par exemple graphes, relations d'ordre partiel, relations d'équivalence, ...), on désire maintenant introduire la combinaison d'éléments d'un ensemble A.

12.1 Opérations binaires

Definition 12.1 Une **opération binaire** $*$ sur un ensemble A est une fonction

$$* \ : \ A \times A \rightarrow A : (a_i, a_j) \rightarrow *(a_i, a_j) = a_i * a_j$$

Par définition, l'opération binaire est une *opération interne*

c.à.d. $a_i * a_j \in A$

Definition 12.2 Une opération binaire est **associative** si $\forall a, b, c \in A$:

$$(a * b) * c = a * (b * c)$$

Definition 12.3 Une opération binaire est **commutative** si $\forall a, b \in A : \ a*b = b * a$.

Example 12.1

$$A = \{0, 1, 2, 3, 4\} = Z^5$$

$$i \otimes j = i + j \pmod 5$$

$\otimes$ *est une opération binaire associative et commutative.*

Example 12.2

$$A = Z$$

$$a * b = a - b$$

$*$ *est une opération binaire qui n'est ni associative ni commutative.*

Example 12.3 $M_2(Z) \stackrel{\Delta}{=}$ *ensemble des matrices* 2×2 *à coefficients entiers.*
$*$ *est la multiplication matricielle habituelle.*
$*$ *est une opération binaire associative et non conmutative*

Definition 12.4 Un élément $e \in A$ est l'**élément neutre** par rapport à l'opération binaire $*$ si $a * e = e * a = a \ \ \forall a \in A$ (il s'appelle aussi *l'identité*).

Souvent nous utilisons 1 ou $1'$ pour représenter le neutre à la place de e.

Example 12.4 *Par rapport à l'exemple 12.3 ci-dessus,*

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

est le neutre de $M_2(Z)$ *par rapport à* $*$ *qui est la multiplication matricielle.*

Definition 12.5 $e_G \in A$ est un **neutre à gauche** si $e_G * a = a \ \forall a \in A$
$e_D \in A$ est un **neutre à droite** si $a * e_D = a \ \forall a \in A$

(Si e_G et e_D existent, alors $e_G = e_D$. Pourquoi?)

Example 12.5

$$A = \{a, b, c\}$$

Considérons la définition suivante de l'opération binaire $*$

$*$	a	b	c
a	a	b	a
b	b	b	b
c	c	b	c

Ici l'entrée en position (1,3) indique que que $a * c = c$.
a et c sont des éléments neutres à droite (car leur colonne est exactement a b c).
Il n'y a pas de neutre à gauche (car il n'y a pas de ligne valant a b c).

12.2 Semi-groupes et Monoïdes

Definition 12.6 Un **semi-groupe** $G = (A, *)$ consiste en un ensemble non vide A muni d'une opération binaire $*$ qui est associative.

Example 12.6 *Le semi-groupe libre généré par* $A = \{a_1 \dots a_n\}$
Soit A^* *l'ensemble de toutes les séquences finies d'éléments de* A*. On définit une opération binaire* $\circ$*, appelée* **concaténation**, $x \circ y =$ *juxtaposition de* x *(à gauche) et de* y
(c.à.d. $x = a_1a_2a_3$ *et* $y = a_3a_2a_1 \Rightarrow x \circ y = a_1a_2a_3a_3a_2a_1$*)*
Cette opération est associative et donc $(A^*, \circ)$ *est un semi-groupe.*
Son neutre est la séquence vide, notée λ*.*

Example 12.7 *Composition de Relations Binaires (Rappel de Chapitre 1)*
Soit $B(S) =$ *l'ensemble de toutes les relations binaires définies sur l'ensemble* S*;*

Soit $\circ$ *une opération binaire telle que la nouvelle relation binaire* $\alpha \circ \beta$ *est définie par:*

$$x(\alpha \circ \beta)y \Longleftrightarrow \exists\ z \in S \text{ t.q. } x\alpha z \text{ et } z\beta y$$

Alors $(B(S), \circ)$ *est un semi-groupe.*

Definition 12.7 Un **monoïde** $M = (A, *)$ est un semi-groupe où il existe un élément neutre $e \in A$ pour l'opération $*$.

Example 12.8 *Quelques Monoïdes*
$(Z_+, +)$ *est un monoïde (neutre* 0*)*
$(Z_+ \backslash \{0\}, \times)$ *est un monoïde (neutre* 1*)*
$(\mathcal{P}(S), \cup)$ *est un monoïde (neutre* $\emptyset$*)*
$(\mathcal{P}(S), \cap)$ *est un monoïde (neutre* S*)*
$(A^*, \circ)$*le semi-groupe libre associaé à* A *est un monoïde (neutre* λ*)*

Definition 12.8 Un semi-groupe (monoïde) $G = (A, *)$ est un **sous semi-groupe (sous monoïde)** d'un autre semi-groupe (monoïde) $G' = (A', \circ)$ si A est un sous-ensemble de A' et $*$ est la restriction de $\circ$ à A. De plus, dans le cas de monoïdes M et M', les neutres doivent coïncider.

Example 12.9 *i)* $(2Z_+, +)$ *est un sousmonoïde de* $(Z_+, +)$*.*

ii) Si $A = \{a_1, a_2, a_3, a_4\}$ *et* $B = \{a_1, a_2\}$*, alors le semi-groupe libre* $(B^*, \circ)$ *est un sousmonoïde de* $(A^*, \circ)$*.*

Definition 12.9 Soit un ensemble fini $S = \{1, \dots, n\}$.
Considérons $M(S)$ l'ensemble de transformations $S \to S$ (Une transformation est une fonction d'un ensemble dans lui-même).
La composée de transformations est associative et un élément neutre pour la

composée existe de façon évidente.
On appelle M le **monoïde de transformations** de S : (transformations $S \to S$ avec $\circ$ composée de fonctions) $M = (M(S), \circ$, Identité des Transformations)

$$(\alpha \circ \beta)(x) = \alpha(\beta(x))$$

Example 12.10

$$S = \{1, 2\}$$

Les transformations sont

$$1_S : \begin{cases} 1 & \to & 1 \\ 2 & \to & 2 \end{cases} ; \alpha : \begin{cases} 1 & \to & 2 \\ 2 & \to & 1 \end{cases} ; \beta : \begin{cases} 1 & \to & 1 \\ 2 & \to & 1 \end{cases} ; \gamma : \begin{cases} 1 & \to & 2 \\ 2 & \to & 2 \end{cases}$$

(Si on autorise toutes les fonctions possibles, l'ordre de $M(S)$ est n^n)

Le tableau des composées est le suivant :

	1_S	α	β	γ
1_S	1_S	α	β	γ
α	α	1_S	γ	β
β	β	β	β	β
γ	γ	γ	γ	γ

Remarquons que $\{\{1_S, \alpha\}, \circ, 1_S\}$ est un sous monoïde (qui est aussi un monoïde de transformations)

12.3 Isomorphisme et Homomorphisme

Comme pour les graphes il est aussi naturel de poser la question si deux monoïdes sont identiques ou semblables.

Definition 12.10 Deux monoïdes $(A, \circ, 1)$ et $(A', *, 1')$ sont **isomorphes** si $\exists$ une bijection $\Phi : A \to A'$ telle que

$$\begin{aligned} \Phi(1) &= 1' \\ \Phi(x \circ y) &= \Phi(x) * \Phi(y) \ \ \forall x, y \in A. \end{aligned}$$

Φ respecte les opérations. Dans ce cas Φ est appelé un **isomorphisme**.

Example 12.11

$$(R, +, 0) \text{ et } (R_+ \backslash \{0\}, \times, 1)$$

Soit

$$\Phi \ : \ R \to R_+ \backslash \{0\} \ : \ x \to \Phi(x) = e^x \text{ (bijection)}$$

$$\Phi(0) = e^0 = 1$$

$$\Phi(x + y) = e^{x+y} = e^x \times e^y = \Phi(x) \times \Phi(y)$$

(Φ est bijective, car Φ^{-1} est une fonction : $\Phi^{-1}(y) = \ln y$ $(y > 0)$).

Cet exemple est evidemment très pratique. Pour multiplier deux nombres, il suffit d'avoir des tables de logarithmes et antilogarithmes (Φ et Φ^{-1}) et d'additioner les logarithmes.

Quand deux monoïdes sont isomorphes, les deux structures algébriques sont indistinctes en ce sens qu'elles ne diffèrent que par les noms utilisés pour définir les éléments des ensembles et les opérations.

Theorem 12.1 Théorème Fondamental des Monoïdes (CAYLEY) *Tout monoïde est isomorphe à un monoïde de transformations.*

Example 12.12
$(Z^4, +_4)$ *(= monoïde car associatif et* 0 *est un neutre)*
c.à.d. l'ensemble des entiers $\{0,1,2,3\}$ *avec l'addition modulo 4.*
Soient les fonctions :

$$0_L : \begin{cases} 0 \to 0 \\ 1 \to 1 \\ 2 \to 2 \\ 3 \to 3 \end{cases} ; \; 1_L : \begin{cases} 0 \to 1 \\ 1 \to 2 \\ 2 \to 3 \\ 3 \to 0 \end{cases} ; \; 2_L : \begin{cases} 0 \to 2 \\ 1 \to 3 \\ 2 \to 0 \\ 3 \to 1 \end{cases} ; \; 3_L : \begin{cases} 0 \to 3 \\ 1 \to 0 \\ 2 \to 1 \\ 3 \to 2 \end{cases}$$

par exemple $1_L(2) = 1 + 2 \pmod 4 = 3$

($\{0_L, 1_L, 2_L, 3_L\}$*, composée de fonctions,* 0_L*) est un monoïde de transformations isomorphe à* $(Z^4, +_4)$*.*

Example 12.13
Les monoïdes $(M(\{1,2\}), \circ, 1_S)$ *et* $(Z^4, +_4)$ *sont-ils isomorphes? (Voir Exemple 12.10)*
Une réponse rapide est non, car le monoïde de transformations n'est pas commutatif.

Un isomorphisme est une propriété forte entre deux monoïdes. On se contente souvent d'une relation plus faible.

Definition 12.11 Etant donnés deux monoïdes $M_1 = (A, \circ, 1)$ et $M_2 = (A', *, 1')$
Une fonction $\Phi : A \to A'$ est un **homomorphisme** du monoïde M_1 vers M_2 si

$$\Phi(x \circ y) = \Phi(x) * \Phi(y) \;\; \forall x, y \in A$$

$$\Phi(1) = 1'$$

(Un isomorphisme est donc un homomorphisme qui est bijectif). $\Phi(A)$ est l'image homomorphe dans A'.

Example 12.14 $M_1 = (\{0,1\}^*$, *concaténation* $\circ, \lambda)$[1]

[1] $\{0,1\}^*$ = séquence de 0 et de 1, λ = séquence nulle

$M_2 = (\{0,1\},\ +_2$ *(addition modulo 2)*,$0)$
M_1 *et* M_2 *sont des monoïdes.*
Pour $x \in \{0,1\}^*$ $\Phi(x) \stackrel{\Delta}{=}$ *nombre de 1 dans* x *(mod 2)*

- Φ *est surjective*
- $\Phi(\lambda) = 0$
- $\Phi(x \circ y) = \Phi(x)\ +_2\ \Phi(y)$

$\rightarrow \Phi$ *est un homomorphisme surjectif (et il est évident que les 2 monoïdes ne sont pas isomorphes).* ∎

Example 12.15 $M_1 = (\{1,2,3\}^*, \circ, \lambda)$.
$M_2 = (\{1,2\}^*, \circ, \lambda)$.
$\Phi_1(x_1, \ldots, x_n) = \alpha(x_1) \ldots \alpha(x_n)$ *où* $\alpha : A \cup \{\lambda\} \rightarrow A \cup \{\lambda\}$ *avec* $\alpha(x_i) = x_i$ *si* $x_i \neq 3, \alpha(3) = \lambda$ *et* $\alpha(\lambda) = \lambda$.

Alors par exemple $\Phi_1(133213112) = 1\lambda\lambda21\lambda112 = 121112$.

Nous avons $\Phi_1(\lambda) = \lambda$, *et*

$$\begin{aligned} \Phi_1(x \circ y) &= \Phi_1(x_1 \ldots x_n y_1, \ldots y_m) \\ &= \alpha(x_1) \ldots \alpha(x_n)\alpha(y_1) \ldots \alpha(y_m) \\ &= \Phi_1(x_1 \ldots x_n) \circ \Phi_1(y_1 \ldots y_m) \\ &= \Phi_1(x) \circ \Phi_1(y) \end{aligned}$$

Donc Φ_1 *est un homomorphisme de* M_1 *à* M_2.
Notons aussi que $\Phi_1^{-1}(\lambda) = (\{3\}^*, \circ, \lambda)$ *est un sousmonoïde de* M_1.

Proposition 12.2 *Si* Φ *est un homomorphisme de* $M_1 = (A, \circ, 1)$ *vers* $M_2 = (A', *, 1')$, *alors* $\Phi(1) = 1'$ *et* $\Phi^{-1}(1')$ *est un sous-monoïde de* M_1 *(appelé le noyau de* Φ*).*

Preuve Nous montrons que $M' = (\Phi^{-1}(1'), \circ, 1)$ est un monoïde M_1.

- Si x et $y \in \Phi^{-1}(1')$, alors $\Phi(x) = \Phi(y) = 1'$
  Donc $\Phi(x \circ y) = \Phi(x) * \Phi(y) = 1' * 1' = 1' \Rightarrow x \circ y \in \Phi^{-1}(1')$
  $\Rightarrow$ opération $\circ$ est interne à $\Phi^{-1}(1')$
- $\circ$ est associative
- et $1 \in \Phi^{-1}(1')$
  $\Rightarrow$ Sous-monoïde
  avec $\circ$ = restriction à $\Phi^{-1}(1')$ de la composition en général. ∎

Proposition 12.3 $(\Phi(A), *, 1')$ *est un sous-monoïde de* $(A', *, 1') = M_2$ *où* Φ *est un homomorphisme entre* $(A, \circ, 1)$ *et* $(A', *, 1')$

Preuve

- Si y_1 et $y_2 \in \Phi(A) \Rightarrow \exists x_1$ et $x_2 \in A$ t.q. $\Phi(x_i) = y_i \;\; i = 1, 2$
  $\Phi(x_1 \circ x_2) = \Phi(x_1) * \Phi(x_2) = y_1 * y_2$,
  et $y_1 * y_2 \in \Phi(A)$ car $x_1 \circ x_2 \in A$.
  En conclusion l'opération de composition restreinte à $\Phi(A)$ est interne.
- $*$ associative
- $1' \in \Phi(A)$ car $\Phi(1) = 1'$

  Alors $(\Phi(A), *, 1')$ est un monoïde, sous-monoïde de M_2.

12.4 Relations de congruence

Soit Φ un homomorphisme de M_1 vers M_2 et $\gamma = \Phi^{-1}\Phi$ la relation binaire définie sur M_1 par

$$x\gamma y \Longleftrightarrow \Phi(x) = \Phi(y)$$

Proposition 12.4 *γ est une relation d'équivalence.*

Preuve La relation binaire est réflexive, symétrique, et transitive.

Definition 12.12 Une **congruence** γ est une relation d'équivalence sur $(S, \circ)$ telle que $\forall a, b, c, d \in S \; : \; a\gamma b$ et $c\gamma d \Rightarrow (a \circ c)\gamma(b \circ d)$

Example 12.16

$$S = (Z_+, +)$$

$$\gamma = \textit{ égalité modulo } n$$

$$si \left\{ \begin{array}{l} p\gamma p' \;\; (p \equiv p' \bmod n) \\ q\gamma q' \;\; (q \equiv q' \bmod n) \end{array} \right\} \; alors \; (p+q)\gamma(p'+q')$$

$\Rightarrow \gamma$ *est une congruence.*

Example 12.17 *Continuons avec l'exemple de monoïde libre (Exemple 12.15).*
$M_1 = (\{1, 2, 3\}^+, \circ, \lambda)$.
$M_2 = (\{1, 2\}^+, \circ, \lambda)$.
$\Phi_1(x_1, \ldots, x_n) = \alpha(x_1) \ldots \alpha(x_n)$ *où* $\alpha : A \cup \{\lambda\} \to A \cup \{\lambda\}$ *avec* $\alpha(x_i) = x_i$ *si* $x_i \neq 3, \alpha(3) = \lambda$.

$(x_1 \ldots x_n)\gamma(y_1 \ldots y_m)$ *si, après suppression des 3's, les séquences qui restent sont les mêmes.*
Par exemple $1333213\gamma 1231$

Proposition 12.5 *Si Φ est un homomorphisme de M_1 vers M_2, la relation binaire $\gamma = \Phi^{-1}\Phi$ est une relation de congruence sur M_1.*

Preuve

On sait déjà que γ est une équivalence.

Si $x_1 \gamma y_1$ et $x_2 \gamma y_2$

Alors $\Phi(x_1) = \Phi(y_1)$ et $\Phi(x_2) = \Phi(y_2)$

Parce que Φ est un homomorphisme,
$$\begin{aligned} \Phi(x_1 \circ x_2) &= \Phi(x_1) * \Phi(x_2) \\ &= \Phi(y_1) * \Phi(y_2) \\ &= \Phi(y_1 \circ y_2) \end{aligned}$$

alorset

$(x_1 \circ x_2)\gamma(y_1 \circ y_2)$ ∎

Les relations de congruence permettent de construire de nouvelles structures algébriques à partir d'une structure algébrique initiale.

Soit γ une relation de congruence définie sur un monoïde $M = (A, \circ, 1)$ (c.à.d. défini sur les éléments de A et l'opération $\circ$).

Comme γ est une équivalence, elle partitionne A en classes d'équivalence. L'ensemble des classes d'équivalence est l'**ensemble quotient** noté A/γ

De plus, comme γ est une congruence

$$x_1 \gamma y_1 \text{ et } x_2 \gamma y_2 \Rightarrow (x_1 \circ x_2)\gamma(y_1 \circ y_2)$$

c.à.d. $x_1 \circ x_2$ et $y_1 \circ y_2$ appartiennent à la même classe d'équivalence.

Soit $E(a)$ la classe d'équivalence de a

On a donc $x_1 \in E(y_1)$ et $x_2 \in E(y_2) \Rightarrow x_1 \circ x_2 \in E(y_1 \circ y_2)$

On peut donc définir une nouvelle opération binaire:

$*$ définie sur les classes d'équivalence :

$E(a) * E(b) \stackrel{\Delta}{=} E(a \circ b)$ (propriété de *classe*).

On vérifie facilement que $M/\gamma = (A/\gamma, *)$ est un monoïde [2]

Definition 12.13 $M/\gamma = (A/\gamma, *)$ est le **sous-monoïde quotient de** M **mod** γ.

Example 12.18

$$\begin{aligned} M &= (Z_+, +, 0) \\ \gamma &= \textit{ égalité modulo 3 : congruence} \\ E(0) &= \{0, 3, 6, \ldots\} \\ E(1) &= \{1, 4, 7, \ldots\} \\ E(2) &= \{2, 5, 8, \ldots\} \\ E(1+2) &= E(0) \rightarrow E(1) * E(2) = E(0) \end{aligned}$$

$*$	$E(0)$	$E(1)$	$E(2)$
$E(0)$	$E(0)$	$E(1)$	$E(2)$
$E(1)$	$E(1)$	$E(2)$	$E(0)$
$E(2)$	$E(2)$	$E(0)$	$E(1)$

[2] $*$ est une opération interne, associative, possédant un neutre sur A/γ

La question suivante à se poser est l'existence d'un homomorphisme du monoïde $M = (A, \circ, 1)$ vers son monoïde quotient M/γ.

Proposition 12.6 *Soit* $\Phi_\gamma(a) \triangleq E(a) =$ *unique classe d'équivalence contenant* a. *Toute relation de congruence* γ *sur un monoïde* M *induit un homomorphisme* Φ_γ *de* M *vers le monoïde quotient* M/γ.

Preuve
Φ_γ est une fonction car $E(a)$ est unique $\forall a$.

$$\begin{aligned} \Phi_\gamma(a \circ b) = E(a \circ b) &= E(a) * E(b) \text{ par définition de } * \\ &= \Phi_\gamma(a) * \Phi_\gamma(b) \\ \Phi_\gamma(1) = E(1) &= \text{neutre pour le monoïde } (M/\gamma, *) \\ \text{car } E(a) * E(1) &= E(a \circ 1) = E(a) \end{aligned}$$

■

Φ_γ est appelé l'**homomorphisme naturel** de M vers M/γ

Nous avons montré que chaque homomorphisme est lié à une congruence, et chaque congruence est associée naturellement à un homomorphisme.

Theorem 12.7 Théorème fondamental sur les homomorphismes entre monoïdes *Si* Φ *est un homomorphisme d'un monoïde* $G_1 = (M, \circ, 1)$ *vers un monoïde* $G_2 = (M', *, 1')$ *et si* γ *est la congruence* $\Phi^{-1}\Phi$ *(c.à.d.* $\forall x, y \in M \;\; x\gamma y \iff \Phi(x) = \Phi(y)$*); alors* $\Phi(G_1) = (\Phi(M), *, 1')$ *est un monoïde (ça on le savait déjà) isomorphe au monoïde quotient* $G_1/\gamma = (M/\gamma, \otimes, E(1))$

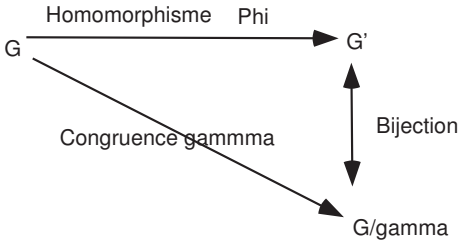


Figure 12.1: Illustration de la preuve du Théorème fondamental

Preuve du théorème fondamental.
Soit $\Psi(E(a)) = \Phi(a)$ (Ψ est bien définie parce que dans une classe d'équivalence associée à γ les valeurs de Φ sont égales).

- $\Phi(M) =$ ensemble des éléments de M' atteints par Φ
  Ψ est donc surjective par définition de $\Phi(M)$
- Si $E(a) \neq E(b)$ et $\Psi(E(a)) = \Psi(E(b)) \Rightarrow \Phi(a) = \Phi(b)$ et $a \in E(b) \rightarrow$ contradiction.
  Ψ est donc injective.

- Ψ est une bijection et, de plus,

$$\begin{array}{rll} \Psi(E(a)\otimes E(b)) & = \Psi(E(a\circ b)) & = \Phi(a\circ b) \\ & & = \Phi(a) * \Phi(b) \\ & & = \Psi(E(a)) * \Psi(E(b)) \\ \Psi(E(1)) & = \Phi(1) & = 1' \end{array}$$

$\Rightarrow \Psi$ est un isomorphisme.

12.5 Groupes et Algèbres

Ici nous introduisons quelques conceptes supplémentaires.

12.5.1 Groupes

Definition 12.14 Dans un monïde $M = (A, \circ, e)$ un élément $a \in A$ a un inverse s'il existe $b \in A$ tel que $a \circ b = b \circ a = e$.

Definition 12.15 Un monoïde $M = (A, \circ, e)$ est un *groupe* si chaque élément de A a un inverse

Example 12.19 $(\{1,2,3,4\}, \times_5, 1)$*, où* $\times_5$ *veut dire multiplication des entiers modulo 5.*

La table de multiplication

$\times_5$	1	2	3	4
1	1	2	3	4
2	2	4	1	3
3	3	1	4	2
4	4	3	2	1

Ici chaque élément a un inverse. Donc c'est un groupe commutatif.

Example 12.20 $A = \{e, a, b, c\}$ *avec* $a^2 = b^2 = c^2 = e, ab = ba = c$*, etc.*

La table de multiplication

	e	a	b	c
e	e	a	b	c
a	a	e	c	b
b	b	c	e	a
c	c	b	a	e

Observons que les vecteurs

$$e = \begin{pmatrix} 0 \\ 0 \end{pmatrix} a = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, b = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, c = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \text{ avec addition } \begin{pmatrix} +_2 \\ +_2 \end{pmatrix}$$

mènent à la même table de multiplication.

12.5.2 Algèbres

Definition 12.16 Une **algèbre universelle** (ou plus simplement une **algèbre**) est un système

$$[S, f_1, f_2, \ldots f_k]$$

qui se compose

- d'un ensemble S;
- de $k \geq 0$ opérations définies sur S t.q. f_i est une opération n-aire sur S.

Example 12.21 *Un treillis $L = \{A, \wedge, \vee\}$ est un algèbre avec deux opérations binaires.*

Example 12.22 *Un treillis distributif $L = \{A, \wedge, \vee, comp, 1, 0\}$ est un algèbre avec deux opérations binaires, une opération unaire (complémentation), et deux opérations 0-aires. Il est appelé un algèbre ou treillis Booléen.*

Definition 12.17 Un **anneau** R est une algèbre $(S, +, \circ)$ où
$(S, +)$ est un groupe abélien, et
$(S, \circ)$ est un semi-groupe, et
$\forall a, b, c \in S \quad a \circ (b + c) = a \circ b + a \circ c$ et $(b + c) \circ a = b \circ a + c \circ a$. ($\circ$ est distributive par rapport à $+$)

Example 12.23
Les entiers pairs $(2Z_+, +, \times)$ avec addition et multiplication forment un algèbre avec deux opérateurs binaires. Cet algèbre est un anneau.

Example 12.24
Les matrices entières 2×2 avec l'addition et la multiplication matricielle forment un algèbre avec deux opérateurs binaires. Cet algèbre est un anneau.