

# Logique

## L3 Informatique

Peter Habermehl

 Université Paris Diderot  
UFR Informatique  
Laboratoire LIAFA  
Peter.Habermehl@liafa.jussieu.fr

## Plan du cours

- 1 Rappels :
  - ▶ **Induction** : ordres bien fondés, définitions inductives, principe d'induction bien fondée, preuves par induction.
  - ▶ **Calcul propositionnel** : syntaxe, sémantique, tables de vérité.
- 2 Systèmes de preuves syntaxiques pour le calcul propositionnel :
  - ▶ Hilbert.
  - ▶ Dédution naturelle.
  - ▶ Gentzen.
  - ▶ Correction et complétude.
- 3 **Calcul des prédicats** :
  - ▶ Syntaxe, sémantique.
  - ▶ Unification et résolution.
  - ▶ Théories équationnelles.

## Modalités du cours

- Chargés de TD :
  - ▶ Thibaut Balabonski, groupe 1, vendredi 14h30-16h30, salle 473F.
  - ▶ Stéphane Zimmermann, groupe 2, lundi 10h30-12h30, salle 470E.
  - ▶ Alexandre Pilkiewicz, groupe 3, mercredi 14h30-16h30, salle 473F.
  - ▶ Marie Ferbus, groupe 4, mercredi 18h30-20h30, salle 478F.
- Examen partiel **obligatoire** : début mars.
- Note 1ère session :  $\frac{1}{2}$  note partiel +  $\frac{1}{2}$  examen final
- Note session rattrapage :  
Max(exam rattrapage,  $\frac{1}{2}$  note partiel +  $\frac{1}{2}$  exam rattrapage)
- Pendant le partiel et les examens, les étudiants auront droit uniquement à la consultation de deux feuilles A4 recto-verso manuscrites et strictement personnelles. Tous les autres documents ne seront pas autorisés.

## Documents du cours

- **Transparents du cours**  
<http://www.liafa.jussieu.fr/~haberm/cours/logique/>
- **Tableau** (exemples et démonstrations)

- **Logique pour l'info. : introduction à la déduction automatique.**  
*S. Cerrito*, VUIBERT.
- **Mathématiques pour l'informatique.**  
*A. Arnold et I. Guessarian*, MASSON.
- **Introduction à la logique.**  
*R. David, K. Nour et C. Raffalli*, DUNOD.
- **Logique et fondements de l'informatique.**  
*R. Lassaigne et M. Rougemont*, HERMES.
- **First-Order Logic and Automated Theorem Proving.**  
*M. Fitting*, SPRINGER.
- **Concrete Mathematics.**  
*R. L. Graham, D. E. Knuth et O. Patashnik*, ADDISON-WESLEY.

- **Logic for Computer Science.**  
*J. Gallier*, WILEY. Disponible en ligne:  
<http://www.cis.upenn.edu/~jean/gbooks/logic.html>
- **Le point aveugle.**  
*J.-Y. Girard*, voir  
<http://iml.univ-mrs.fr/~girard/cours/cours.html>
- **Logicomics.**  
*A. Doxiadis, C. Papadimitriou, A. Papadatos, A. Di Donna*, VUIBERT.

---

## Notions préliminaires

---

## Ensembles

**Définition :** Soient deux ensembles  $\mathcal{A}, \mathcal{B}$  inclus dans  $\mathcal{U}$  (Univers).  
L'**intersection** de  $\mathcal{A}$  et  $\mathcal{B}$  est  $\mathcal{A} \cap \mathcal{B} = \{e \in \mathcal{U} \mid e \in \mathcal{A} \text{ et } e \in \mathcal{B}\}$   
L'**union** de  $\mathcal{A}$  et  $\mathcal{B}$  est  $\mathcal{A} \cup \mathcal{B} = \{e \in \mathcal{U} \mid e \in \mathcal{A} \text{ ou } e \in \mathcal{B}\}$   
La **différence** de  $\mathcal{A}$  et  $\mathcal{B}$  est  $\mathcal{A} \setminus \mathcal{B} = \{e \in \mathcal{U} \mid e \in \mathcal{A} \text{ et } e \notin \mathcal{B}\}$   
Le **complémentaire** de  $\mathcal{A}$  est  $\overline{\mathcal{A}} = \mathcal{U} \setminus \mathcal{A} = \{e \in \mathcal{U} \mid e \notin \mathcal{A}\}$   
 $\mathcal{P}(\mathcal{A})$  est l'ensemble de toutes les **parties** (sous-ensembles) de l'ensemble  $\mathcal{A}$ .



(Lois de *de Morgan*)

$$\mathcal{A} \cup \mathcal{B} = \overline{\overline{\mathcal{A}} \cap \overline{\mathcal{B}}}$$

$$\overline{\mathcal{A} \cap \mathcal{B}} = \overline{\mathcal{A}} \cup \overline{\mathcal{B}}$$

**Définition :** Le **produit cartésien** de  $n$  ensembles  $\mathcal{A}_1 \dots \mathcal{A}_n$  est l'ensemble de  $n$ -uplets  $\mathcal{A}_1 \times \dots \times \mathcal{A}_n = \{(a_1, \dots, a_n) \mid a_i \in \mathcal{A}_i\}$ . Si  $\mathcal{A}_i = \mathcal{A}$  pour tout  $i$ , on note  $\mathcal{A}^n$  le produit  $\mathcal{A}_1 \times \dots \times \mathcal{A}_n$ .

**Définition :** Une **relation n-aire** sur  $\mathcal{A}_1 \dots \mathcal{A}_n$  est un sous-ensemble de  $\mathcal{A}_1 \times \dots \times \mathcal{A}_n$ .

**Définition :** Soit  $R \subseteq \mathcal{A} \times \mathcal{A}$  une relation **binaires**.

- $R$  est **réflexive** ssi pour tout  $x \in \mathcal{A}$ ,  $(x, x) \in R$ .  
 $R$  est **irréflexive** ssi pour tout  $x \in \mathcal{A}$ ,  $(x, x) \notin R$ .
- $R$  est **symétrique** si pour tout  $x, y \in \mathcal{A}$ ,  $(x, y) \in R$  implique  $(y, x) \in R$ .  
 $R$  est **anti-symétrique** si pour tout  $x, y \in \mathcal{A}$ ,  $(x, y) \in R$  et  $(y, x) \in R$  implique  $x = y$ .
- $R$  est **transitive** si pour tout  $x, y, z \in \mathcal{A}$ ,  $(x, y) \in R$  et  $(y, z) \in R$  implique  $(x, z) \in R$ .

## Exemples

**Exemple :** La relation  $\geq$  sur les entiers naturels est réflexive, la relation  $>$  sur les entiers naturels est irréflexive.

**Exemple :** La relation  $=$  sur les ensembles est symétrique, la relation  $\subseteq$  sur les entiers naturels est anti-symétrique.

**Exemple :** La relation  $\supseteq$  sur les ensembles est transitive.

- $(x, y) \in R$  peut s'écrire aussi  $x R y$ .
- On peut utiliser un symbole à la place de  $R$ :  
Ainsi par exemple, si  $\leq$  est une relation, alors  $(x, y) \in \leq$  s'écrit  $x \leq y$ .
- On écrit  $y \geq x$  lorsque  $x \leq y$ .

## Équivalence et Congruence

**Définition :**

- $R$  est une **équivalence** si elle est réflexive, symétrique et transitive.

**Exercice :** Montrer que  $\sim = \{(x, y) \mid 3 \text{ est diviseur de } x - y\}$  est une équivalence.

- $R$  est une **congruence** p.r. à  $f$  si  $R$  est une équivalence compatible avec  $f$ , c'est à dire, si  $a_1 R b_1 \dots a_n R b_n$  implique  $f(a_1, \dots, a_n) R f(b_1 \dots b_n)$ .

**Exercice :** Montrer que  $\sim = \{(x, y) \mid 3 \text{ est diviseur de } x - y\}$  est une congruence par rapport à  $+$  et à  $*$ .

La **classe d'équivalence** de  $a \in \mathcal{A}$  par rapport à une équivalence  $R$  est l'ensemble  $[a]_R = \{b \in \mathcal{A} \mid aRb\}$ .

## Les clôtures

**Définition :** La **clôture transitive** d'une relation  $\mathcal{R}$  est donnée par

$$\mathcal{R}^+ = \bigcup_{n=1}^{\infty} \mathcal{R}^n$$

La **clôture réflexive et transitive** d'une relation  $\mathcal{R}$  est donnée par

$$\mathcal{R}^* = \bigcup_{n=0}^{\infty} \mathcal{R}^n = \mathcal{R}^+ \cup \mathcal{R}^0$$

**Exemple :** Dans l'exemple d'avant,  $\mathcal{R}^* = A \times A$ .

# Composition de relations

**Définition :** Si  $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{B}$  et  $\mathcal{S} \subseteq \mathcal{B} \times \mathcal{C}$ , alors la **composition** de  $\mathcal{S}$  avec  $\mathcal{R}$  est une relation dans  $\mathcal{A} \times \mathcal{C}$  t.q.

$$\mathcal{S} \circ \mathcal{R} = \{(x, y) \in \mathcal{A} \times \mathcal{C} \mid \exists z \in \mathcal{B} (x, z) \in \mathcal{R} \text{ et } (z, y) \in \mathcal{S}\}.$$

**Définition :** Soit  $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ . On note  $\mathcal{R}^n$  la **n-composition** de  $\mathcal{R}$  avec elle-même définie par récurrence comme suit :

$$\begin{aligned} \mathcal{R}^0 &= \{(a, a) \mid a \in \mathcal{A}\} \\ \mathcal{R}^{n+1} &= \mathcal{R}^n \circ \mathcal{R} = \mathcal{R} \circ \mathcal{R}^n = \underbrace{\mathcal{R} \circ \dots \circ \mathcal{R}}_{n+1 \text{ fois}} \end{aligned}$$

**Exemple :** Soit  $A = \{Paris, Lyon, Toulouse\}$  et  $R = \{(Paris, Lyon), (Paris, Toulouse), (Lyon, Paris), (Toulouse, Paris)\}$ ,  $R^2 = \{(Paris, Paris), (Lyon, Lyon), (Toulouse, Toulouse), (Lyon, Toulouse), (Toulouse, Lyon)\}$ , Calculer  $R^3$ .

## Fonctions

**Définition :** Une **fonction  $f$  entre deux ensembles  $\mathcal{A}$  et  $\mathcal{B}$** , notée  $f : \mathcal{A} \rightarrow \mathcal{B}$ , est une relation sur  $\mathcal{A} \times \mathcal{B}$  t.q. pour tout  $x, y, z$  si  $(x, y) \in f$  et  $(x, z) \in f$ , alors  $y = z$ .

**Notation :** On écrit  $f(x)$  pour dénoter l'**unique** élément  $y$  t.q.  $(x, y) \in f$  et  $f(\mathcal{C}) = \{y \in \mathcal{B} \mid \exists x \in \mathcal{C}, f(x) = y\}$ .

On note  $id_{\mathcal{A}}$  la fonction **identité** sur  $\mathcal{A}$  donnée par  $id_{\mathcal{A}}(x) = x$ .

**Définition :** Soit  $f : \mathcal{A} \rightarrow \mathcal{B}$  une fonction.

- Le **domaine** de  $f$  est  $Dom(f) = \{x \in \mathcal{A} \mid \exists y \in \mathcal{B}, (x, y) \in f\}$
- L'**image** de  $f$  est  $Im(f) = \{y \in \mathcal{B} \mid \exists x \in \mathcal{A}, (x, y) \in f\}$
- L'**inverse** (pas toujours une fonction) de  $f$  est  $f^{-1} = \{(y, x) \in \mathcal{B} \times \mathcal{A} \mid (x, y) \in f\}$

## Composition de fonctions

### Définition :

- La **composition** de  $f : \mathcal{B} \rightarrow \mathcal{C}$  avec  $g : \mathcal{A} \rightarrow \mathcal{B}$  est la fonction  $f \circ g : \mathcal{A} \rightarrow \mathcal{C}$ , où  $f \circ g(x) = f(g(x))$ .

**Exemple :**  $f(x) = x^2$ ,  $g(x) = x + 4$ ,  $f \circ g(x) = (x + 4)^2$ ,  
 $g \circ f(x) = x^2 + 4$ .

- La **n-composition** de  $f$  avec **elle-même**, notée  $f^n$ , est défini par récurrence sur  $n$  :

- ▶ Si  $n = 0$ , alors  $f^0 = id$
- ▶ Si  $n > 0$ , alors  $f^n = f \circ f^{n-1}$

**Exemple :**  $f(x) = x + 2$ ,  $f^0(x) = x$ ,  $f^1(x) = x + 2$ ,  $f^2(x) = x + 4$ ,  
 $f^3(x) = x + 6$ , ...,  $f^n(x) = x + 2.n$ .

**Exercice :** Soit  $n > 0$ . Montrer que  $f^n = f^{n-1} \circ f$ .

## Propriétés des fonctions

**Définition :** Une fonction  $f : \mathcal{A} \rightarrow \mathcal{B}$  est **injective** ssi pour tout  $x, y \in \mathcal{A}$ ,  $f(x) = f(y)$  implique  $x = y$ .

**Exemple :**  $f(x) = x + 2$  sur les entiers est injective.  $f(x) = x \setminus \{3\}$  sur les ensembles d'entiers n'est pas injective. Ainsi  $f(\{2, 3, 4\}) = f(\{2, 4\})$  mais  $\{2, 3, 4\} \neq \{2, 4\}$ .

**Définition :** Une fonction  $f : \mathcal{A} \rightarrow \mathcal{B}$  est **surjective** ssi pour tout  $y \in \mathcal{B}$  il existe  $x \in \mathcal{A}$  tel que  $f(x) = y$ .

**Exemple :**  $f(x) = x \text{ div } 2$  sur les entiers naturels est surjective.  $f(x) = x + 2$  sur les entiers naturels n'est pas surjective.

**Définition :** Une fonction est **bijective** ssi elle est injective et surjective.

**Exemple :** Soit  $\mathcal{A}$  l'ensemble de mots de longueur 3 contenant uniquement 0 et 1. Soit  $\mathcal{B} = \{0 \dots 7\}$ . Soit  $f("b_2b_1b_0") = b_2.2^2 + b_1.2^1 + b_0.2^0$ . Cette fonction est injective et surjective, donc bijective.

## Préordres, ordres

### Définition :

- Un **préordre** est une relation réflexive et transitive.

**Exemple :**  $\mathcal{R} = \{(2, 2), (3, 3), (4, 4), (3, 2), (2, 3), (2, 4), (3, 4)\}$ .

- Un **ordre** ou **ordre partiel** est une relation réflexive, anti-symétrique et transitive.

**Notation :**  $\geq$

**Exemple :**  $\mathcal{R}$  n'est pas un ordre car  $(3, 2), (2, 3)$  mais  $2 \neq 3$ .  
 $\mathcal{S} = \{(2, 2), (3, 3), (4, 4), (2, 3), (2, 4), (3, 4)\}$  est un ordre.

## Ordre stricte

**Définition :** Un **ordre strict** est une relation irreflexive et transitive.

**Notation :**  $>$

**Exemple :**  $>$  sur les entiers,  $\supset$  sur les ensembles.

**Définition :** Un ordre strict est **bien fondé** ssi il n'existe aucune chaîne infinie décroissante (i.e., de la forme  $a_0 > a_1 > a_2 > \dots$ ).

**Exemple :**  $>$  sur les entiers naturels est bien fondé.  $>$  sur tous les entiers n'est pas bien fondé.  $\supset$  sur les ensembles est bien fondé.

## Majorants/minorants et bornes supérieures/inférieures

Soit  $\mathcal{E}$  un ensemble muni d'un ordre  $\leq$ . Soit  $\mathcal{A} \subseteq \mathcal{E}$ .

### Définition :

Un **majorant** de  $\mathcal{A}$  est un  $x \in \mathcal{E}$  t.q. pour tout  $y \in \mathcal{A}$ ,  $y \leq x$ .

Un **minorant** de  $\mathcal{A}$  est un  $x \in \mathcal{E}$  t.q. pour tout  $y \in \mathcal{A}$ ,  $x \leq y$ .

La **borne supérieure** de  $\mathcal{A}$ , notée  $\sup(\mathcal{A})$ , est le plus petit des majorants de  $\mathcal{A}$  (si  $z$  est un majorant de  $\mathcal{A}$  alors  $\sup(\mathcal{A}) \leq z$ ).

La **borne inférieure** de  $\mathcal{A}$ , notée  $\inf(\mathcal{A})$ , est le plus grand des minorants de  $\mathcal{A}$  (si  $z$  est un minorant de  $\mathcal{A}$  alors  $z \leq \inf(\mathcal{A})$ ).

**Exemple :** Soit  $\mathcal{A} = \{1, \dots, 10\}$ . Tous les entiers dans  $\{10, \dots\}$  sont des majorants de  $\mathcal{A}$  et 10 est la borne supérieure.

**Exemple :** Si  $a \leq c$ ,  $a \leq d$ ,  $b \leq c$ ,  $b \leq d$ , alors  $a$  et  $b$  sont des minorants, mais comme ils sont incomparables il n'y a pas de borne inférieure.

**Exemple :** Si  $E_i$  sont des ensembles dans  $\mathcal{P}(\mathcal{E})$ , alors le sup est  $\bigcup_i E_i$  et le inf est  $\bigcap_i E_i$ .

---

## Définitions Inductives et preuves par induction

---

## Définitions inductives en informatique

- Syntaxe concrète
- Syntaxe abstraite
- Règles de typage
- Règles d'évaluation

## Le principe

Une définition inductive est caractérisée par :

- Une ou plusieurs **assertions**
- Un ensemble de **règles** d'inférence pour dériver ces assertions

### Exemple :

- Assertion : "X est naturel" ou "X nat"
- Règles d'inférence :
  - R1: 0 est naturel
  - R2: Si n est naturel, alors succ(n) est naturel.

## Notation

Les règles d'inférence sont notées

$$\frac{\text{Hypothèse}_1 \dots \text{Hypothèse}_n}{\text{Conclusion}} \text{ (Nom de la règle)}$$

- Conclusion est une assertion
- Hypothèse<sub>1</sub> ... Hypothèse<sub>n</sub> sont des assertions
- En général  $n \geq 0$ . Si  $n = 0$  la règle est un **axiome**

## Exemple (règle binaire)

Les arbres binaires

$$\frac{}{\text{vide est un arbre binaire}} \text{ (Abin-nil)}$$

$$\frac{A_1 \text{ est un arbre binaire} \quad A_2 \text{ est un arbre binaire}}{\text{node}(A_1, A_2) \text{ est un arbre binaire}} \text{ (Abin-ind)}$$

## Exemple (règle unaire)

Les entiers naturels

$$\frac{}{0 \text{ est naturel}} \text{ (Nat0)} \quad \frac{n \text{ est naturel}}{\text{succ}(n) \text{ est naturel}} \text{ (Nat+)}$$

## Exemple

Les mots sur un alphabet A

$$\frac{}{\epsilon \text{ mot}} \quad \frac{a \in A \quad n \text{ mot}}{a.n \text{ mot}}$$



Un ensemble inductif est le plus petit ensemble engendré par un système de règles d'inférence.

## Preuves par induction

- Induction sur les entiers
  - ▶ Induction mathématique
  - ▶ Induction complète
  - ▶ Équivalence
- Induction bien fondée
- Induction structurelle
- Induction sur un ensemble inductif

---

## Preuves par Induction

---

## Induction sur les entiers I (induction mathématique)

**Théorème :** Soit  $P$  une propriété sur les entiers. Supposons

$$(IM1) P(0),$$

$$(IM2) \forall n \in \mathbf{N}. P(n) \rightarrow P(n+1),$$

alors  $\forall n \in \mathbf{N}. P(n)$

$$1) \sum_{i=1}^n i = \frac{n * (n + 1)}{2} \quad 2) \quad n^2 = \sum_{i=1}^n (2i - 1)$$

Mais comment prouver

- ① “Tout entier est décomposable en produit de nombres premiers”
- ② “Si  $n$  est divisible par 3, alors  $fib(n)$  est pair, sinon  $fib(n)$  est impair”.

**Théorème :** Soit  $P$  une propriété sur les entiers. Supposons

$$(IC) \quad \forall n \in \mathbf{N}. ((\forall k < n \rightarrow P(k)) \rightarrow P(n))$$

alors  $\forall n \in \mathbf{N}. P(n)$

## Équivalence des deux principes

Malgré l'apparente supériorité du deuxième principe, on prouve

**Théorème :** Induction mathématique et complète sont équivalentes.

## Un théorème fondamental

**Théorème :** Tous les français sont d'accord avec le Président de la République.

**Preuve :** On montre, par induction sur le nombre de français, que tout groupe de  $n$  personnes contenant le Président est d'accord avec lui.

Cas de base: il y a seulement le Président, trivial.

Cas inductif: on suppose l'énoncé vrai pour tout groupe de  $n$  personnes, et on le prouve pour tout groupe de  $n + 1$ .

Numérotons de 1 à  $n + 1$  les personnes en question, de façon que le Président soit le numéro  $n$ , et considérons le groupe  $A$  des premières  $n$  et le groupe  $B$  des dernières  $n$  personnes.

Les deux groupes contiennent le Président et sont de taille  $n < n + 1$ . On peut donc appliquer l'hypothèse d'induction et en déduire qu'ils sont tous d'accord avec le Président (qui est dans les deux), ce qui nous permet de conclure.

**vrai ou faux?**

## Principe d'induction bien fondée

Un ensemble  $\mathcal{A}$ , un ordre strict  $>$  et une propriété  $P$  sur  $\mathcal{A}$

**Principe d'induction :**

Si

- 1 "pour tout élément minimal  $y \in \mathcal{A}$  on a  $P(y)$ "
- 2 "le fait que  $P(z)$  soit vérifiée pour **tout** élément  $z < x$  implique  $P(x)$ "

alors

"pour tout  $x \in \mathcal{A}$  on a  $P(x)$ "

## Exemples

- **Les mots :**  
 $P(m)$  est la propriété :  
 $\text{concat}(\text{concat}(m, v_1), v_2) = \text{concat}(m, \text{concat}(v_1, v_2))$
- **Les arbres binaires :**  
 $P(a)$  est la propriété :  $\text{feuilles}(a) = \text{noeuds\_internes}(a) + 1$

## Ce principe est-il toujours bien défini?

Soit  $>$  un ordre strict.

**Théorème :**

Si  $>$  est **bien fondé**, alors le principe d'induction est correct.

**Théorème :**

Si le principe d'induction est correct, alors  $>$  est **bien fondé**.

**Corollaire :** Le principe d'induction est correct pour les ensembles inductifs.

**Corollaire :** Le principe d'induction structurelle est correct.

## Induction sur quelques ordres bien fondés

- Ordre lexicographique
- Ordre multi-ensemble
- Combinaisons

## Ordres lexicographiques

Soit  $>_{A_i}$  un ordre strict sur l'ensemble  $\mathcal{A}_i$ .

**Ordre lexicographique sur le produit de 2 ensembles:**

$$(x, y) >_{lex} (x', y') \text{ ssi } (x >_{A_1} x') \text{ ou } (x = x' \text{ et } y >_{A_2} y')$$

**Exemple :**

$$(4, "abc") >_{lex} (3, "abc") >_{lex} (2, "abcde") >_{lex} (2, "bcde") >_{lex} (2, "e") >_{lex} (1, "e") >_{lex} (0, \epsilon)$$

## Ordre lexicographique sur le produit de $n$ ensembles

Si chaque  $>_{A_i}$  est un ordre strict sur l'ensemble  $\mathcal{A}_i$ , alors  $>_{lex}$  est un ordre strict qui permet de comparer deux  $n$ -uplets de la manière suivante:

$$(x_1, \dots, x_n) >_{lex} (x'_1, \dots, x'_n) \text{ ssi } \begin{array}{l} \exists 1 \leq j \leq n \\ (x_j >_{A_j} x'_j \text{ and } \forall 1 \leq i < j \ x_i = x'_i) \end{array}$$

**Théorème :** Si chaque  $>_{A_i}$  est un ordre strict bien fondé sur  $\mathcal{A}_i$ , alors l'ordre lexicographique  $>_{lex}$  sur le produit de  $\mathcal{A}_1 \times \dots \times \mathcal{A}_n$  est un ordre strict bien fondé sur  $\mathcal{A}_1 \times \dots \times \mathcal{A}_n$ .

**Avertissement :**  $>_{lex}$  n'est pas l'ordre du dictionnaire!!

## Exemple : la fonction d'Ackermann



Montrer par induction que la fonction suivante termine.

$$\begin{array}{lcl} \text{Ackermann}(0, n) & = & n+1 \\ \text{Ackermann}(m+1, 0) & = & \text{Ackermann}(m, 1) \\ \text{Ackermann}(m+1, n+1) & = & \text{Ackermann}(m, \text{Ackermann}(m+1, n)) \end{array}$$

## Les multi-ensembles

**Définition :** Soit  $\mathcal{A}$  un ensemble. Un **multi-ensemble** de base  $\mathcal{A}$  est une fonction  $\mathcal{M} : \mathcal{A} \rightarrow \mathbf{N}$ . Le multi-ensemble  $\mathcal{M}$  est **fini** si  $\mathcal{M}(x) > 0$  seulement pour un nombre fini d'éléments de  $\mathcal{A}$ .

**Notation :**  $\{\{a, a, b\}\}$ .

**Définition :**  $\mathcal{M} \succ_{mul} \mathcal{N}$  ssi  $\mathcal{N}$  s'obtient à partir de  $\mathcal{M}$  en appliquant la règle suivante un nombre fini de fois : enlever un élément  $x$  de  $\mathcal{M}$  et le remplacer par un nombre fini d'éléments plus petits que  $x$  (par rapport à l'ordre  $>$ ).

**Notation :**  $\{\{5, 3, 1, 1\}\}$

**Exemple :**  $\{\{5, 3, 1, 1\}\} \succ_{mul} \{\{4, 3, 3, 1\}\}$ .

Car  $\{\{5, 3, 1, 1\}\} \succ_{mul} \{\{4, 3, 3, 1, 1\}\} \succ_{mul} \{\{4, 3, 3, 1\}\}$

**Théorème :** Si  $>_{\mathcal{A}}$  est un ordre strict bien fondé sur  $\mathcal{A}$ , alors  $>_{mul}$  est un ordre strict bien fondé sur les multi-ensembles de base  $\mathcal{A}$ .

Un homme possède une somme d'argent en euros. Chaque jour il procède de la façon suivante:

- soit il jette une pièce de monnaie dans une fontaine,
- ou bien il change l'un de ses billets à la banque par un nombre arbitraire de pièces de monnaie de valeur quelconque.

Montrer que ce processus termine, c'est à dire, que dans un temps fini l'homme est ruiné.

---

## Le calcul propositionnel

---

## Logique Propositionnelle

- Syntaxe
- Sémantique
- Définissabilité
- Systèmes de preuves
  - ▶ Systèmes de preuves sémantiques (tables de vérité)
  - ▶ Systèmes de preuves syntaxiques

## Syntaxe de la logique propositionnelle

Soit  $\mathcal{R}$  un ensemble dénombrable de lettres dites **propositionnelles**.

**Définition :** L'ensemble de **formules** de la logique propositionnelle est le plus petit ensemble contenant  $\mathcal{R}$  et fermé par les opérations binaires  $\vee$ ,  $\wedge$ ,  $\rightarrow$  et l'opération unaire  $\neg$ .

**Exemple :**  $\neg(p)$        $\vee(p, p)$        $\rightarrow(\wedge(p, q), \neg(r))$   
**Autre notation :**  $\neg p$        $p \vee p$        $(p \wedge q) \rightarrow \neg r$

**Notation :** On écrira  $\#$  pour  $\vee$ ,  $\wedge$  ou  $\rightarrow$ .

**Remarque :** C'est un ensemble inductif, donc on pourra appliquer le principe d'induction.

## $\mathcal{SF}(A)$ : sous-formules d'une formule $A$

- Si  $A$  est une lettre  $p$ ,  $\mathcal{SF}(A) = \{p\}$ .
- Si  $A$  est  $\neg B$ ,  $\mathcal{SF}(A) = \{\neg B\} \cup \mathcal{SF}(B)$ .
- Si  $A$  est  $B \# C$ ,  $\mathcal{SF}(A) = \{B \# C\} \cup \mathcal{SF}(B) \cup \mathcal{SF}(C)$ .

## Sémantique de la logique propositionnelle

Étant donnée une valeur de l'ensemble  $\mathbf{BOOL} = \{\mathbf{V}, \mathbf{F}\}$  pour chaque lettre propositionnelle, on veut établir la valeur d'une formule propositionnelle  $A$ .

- Fixer une **interprétation**  $I : \mathcal{R} \rightarrow \mathbf{BOOL}$  qui donne  $\mathbf{V}$  ou  $\mathbf{F}$  à chaque lettre propositionnelle.
- Définir la **fonction booléenne unaire**  $\mathcal{FB}_{\neg} : \mathbf{BOOL} \rightarrow \mathbf{BOOL}$  et les **fonctions booléennes binaires**  $\mathcal{FB}_{\vee}, \mathcal{FB}_{\wedge}, \mathcal{FB}_{\rightarrow} : \mathbf{BOOL}^2 \rightarrow \mathbf{BOOL}$ .
- Construire la **valeur de vérité** de la formule  $A$ .

$$\begin{aligned}\mathcal{FB}_{\neg}(\mathbf{V}) &= \mathbf{F} \\ \mathcal{FB}_{\neg}(\mathbf{F}) &= \mathbf{V}\end{aligned}$$

$$\begin{array}{ll}
 \mathcal{FB}_\vee(\mathbf{V}, \mathbf{V}) = \mathbf{V} & \mathcal{FB}_\wedge(\mathbf{V}, \mathbf{V}) = \mathbf{V} \\
 \mathcal{FB}_\vee(\mathbf{V}, \mathbf{F}) = \mathbf{V} & \mathcal{FB}_\wedge(\mathbf{V}, \mathbf{F}) = \mathbf{F} \\
 \mathcal{FB}_\vee(\mathbf{F}, \mathbf{V}) = \mathbf{V} & \mathcal{FB}_\wedge(\mathbf{F}, \mathbf{V}) = \mathbf{F} \\
 \mathcal{FB}_\vee(\mathbf{F}, \mathbf{F}) = \mathbf{F} & \mathcal{FB}_\wedge(\mathbf{F}, \mathbf{F}) = \mathbf{F} \\
 \\ 
 \mathcal{FB}_\neg(\mathbf{V}, \mathbf{V}) = \mathbf{V} & \\
 \mathcal{FB}_\neg(\mathbf{V}, \mathbf{F}) = \mathbf{F} & \\
 \mathcal{FB}_\neg(\mathbf{F}, \mathbf{V}) = \mathbf{V} & \\
 \mathcal{FB}_\neg(\mathbf{F}, \mathbf{F}) = \mathbf{V} & 
 \end{array}$$

- Si  $A$  est une lettre  $p$ ,  $[A]_I = I(p)$ .
- Si  $A$  est  $\neg B$ ,  $[A]_I = \mathcal{FB}_\neg([B]_I)$ .
- Si  $A$  est  $B \# C$ ,  $[A]_I = \mathcal{FB}_\#([B]_I, [C]_I)$ .

**Exercice :** Soit  $I$  l'interprétation  $I(p) = \mathbf{V}$ ,  $I(q) = \mathbf{F}$ . Calculer la valeur de vérité de la formule  $(p \vee q) \rightarrow \neg(q \wedge q)$  par rapport à  $I$ .

## Tables de vérité

**À quoi ça sert?** Méthode pour raisonner sur les modèles de formules propositionnelles.

**Comment ça marche?** Soit  $A$  une formule ayant comme lettres propositionnelles l'ensemble  $\{p_1, \dots, p_n\}$  et dont l'ensemble de sous-formules est  $\{A_1, \dots, A_k\}$ .

- 1 Construire une table où chaque colonne est étiquetée par une lettre  $p_i$  ou bien par une sous-formule  $A_j$ .
- 2 Pour chaque ligne  $m$  de la table :
  - 1 Donner une interprétation  $I_m$  aux lettres  $p_1, \dots, p_n$ .
  - 2 Calculer les valeurs  $[A_1]_{I_m}, \dots, [A_k]_{I_m}$

## Satisfaire et falsifier une formule

Soit  $I$  une interprétation,  $A$  une formule et  $\Delta$  un ensemble de formules.

**Définition :**

$I$  **satisfait** une **formule**  $A$  si  $[A]_I = \mathbf{V}$

$I$  **falsifie** une **formule**  $A$  si  $[A]_I = \mathbf{F}$ .

$I$  **satisfait** un **ensemble de formules**  $\Delta$  si  $I$  satisfait toute formule de  $\Delta$ .

$I$  **falsifie** un **ensemble de formules**  $\Delta$  ssi il existe au moins une formule  $A$  dans  $\Delta$  telle que  $[A]_I = \mathbf{F}$ .

**Définition :** Une formule  $A$  est **satisfaisable** s'il existe au moins une interprétation  $I$  qui satisfait  $A$ . Un **ensemble de formules**  $\Delta$  est **satisfaisable** s'il existe au moins une interprétation  $I$  telle que  $I$  satisfait  $\Delta$ , c'est à dire s'il existe au moins une interprétation  $I$  telle que  $I$  satisfait toutes les formules de  $\Delta$  en même temps.

**Définition :** Une formule  $A$  est **contradictoire** ou **insatisfaisable** si elle n'est pas satisfaisable, c'est à dire s'il n'existe pas d'interprétation  $I$  qui satisfait  $A$  (si toute interprétation falsifie  $A$ ).  
Un **ensemble de formules**  $\Delta$  est **contradictoire** ou **insatisfaisable** si il n'est pas satisfaisable (s'il n'existe pas d'interprétation qui satisfait toutes les formules de  $\Delta$  en même temps).

**Définition :** Une formule  $A$  est **valide** si toute interprétation satisfait  $A$ .  
Un **ensemble** de formules  $\Delta$  est **valide** si toute formule de  $\Delta$  est valide.

**Définition :** Une formule  $A$  est **conséquence logique** d'un **ensemble de formules**  $\Delta$ , noté  $\Delta \models A$ , si toute interprétation qui satisfait  $\Delta$  satisfait aussi  $A$ .

## Comment lire une table de vérité?

- Si la colonne étiquetée par la formule  $A$  (qui est une sous-formule de  $A$ ) ne contient que de **V**, alors  $A$  est **valide**.
- Si la colonne de la formule  $A$  ne contient que de **F**, alors  $A$  est **contradictoire**.
- Sinon, l'interprétation qui rends **V** la colonne de la formule  $A$  **satisfait**  $A$  et l'interprétation qui rends **F** la colonne de la formule  $A$  **falsifie**  $A$ .

## Équivalence logique

**Définition :** Deux formules  $A$  et  $B$  sont **équivalentes**, noté  $A \equiv B$ , ssi  $\{A\} \models B$  et  $\{B\} \models A$ .

**Remarque :**  $A \equiv B$  ssi  $(A \rightarrow B) \wedge (B \rightarrow A)$  est valide.

## Encore quelques exemples

(Associativité)	$(A \vee B) \vee C \equiv A \vee (B \vee C)$
	$(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$
(Commutativité)	$A \vee B \equiv B \vee A$
	$A \wedge B \equiv B \wedge A$
(Idempotence)	$A \vee A \equiv A$
	$A \wedge A \equiv A$
(Lois de De Morgan)	$\neg(A \wedge B) \equiv \neg A \vee \neg B$
	$\neg(A \vee B) \equiv \neg A \wedge \neg B$
(Distributivité)	$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
	$A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$
(Loi de la double négation)	$\neg\neg A \equiv A$
(Définissabilité de $\rightarrow$ )	$A \rightarrow B \equiv \neg A \vee B$

## Remarques

- 1  $\{E_1, \dots, E_n\} \models A$  ssi la formule  $E_1 \wedge \dots \wedge E_n \rightarrow A$  est valide pour  $n \geq 1$ .
- 2 L'ensemble vide est satisfaisable.
- 3 Toute formule valide est conséquence logique d'un ensemble quelconque de formules, en particulier de l'ensemble vide.
- 4  $\emptyset \models A$  ssi la formule  $A$  est valide.
- 5 Si  $\Delta$  est satisfaisable et  $\Gamma \subseteq \Delta$ , alors  $\Gamma$  est satisfaisable.
- 6 L'ensemble de toutes les formules est contradictoire.
- 7 Si  $\Delta$  est satisfaisable, alors  $\Delta$  est finiment satisfaisable.
- 8 Si  $\Gamma$  est contradictoire et  $\Gamma \subseteq \Delta$ , alors  $\Delta$  est contradictoire.
- 9 Toute formule est conséquence logique d'un ensemble insatisfaisable de formules.
- 10  $A$  est valide ssi  $\neg A$  est insatisfaisable.
- 11  $\Delta \models A$  ssi  $\Delta \cup \{\neg A\}$  est insatisfaisable.

## Théorème de compacité

**Théorème :** Un ensemble de formules  $\Delta$  est satisfaisable ssi tout sous-ensemble fini de  $\Delta$  est satisfaisable.