

# Logique

## L3 Informatique

Peter Habermehl



Université Paris Diderot, Sorbonne Paris Cité

UFR Informatique

Laboratoire LIAFA

`Peter.Habermehl@liafa.univ-paris-diderot.fr`

# Modalités du cours

- Chargés de TD (début la semaine prochaine) :
  - ▶ Groupe 1:  
Amélie Gheerbrant, Mercredi 10h30-12h30. Salle 1009, Sophie Germain
  - ▶ Groupe 2:  
Vlady Ravelomanana, Lundi 10h30-12h30, Salle 2035, Sophie Germain
  - ▶ Groupe 3:  
Arnaud Sangnier, Mercredi 13h30-15h30. Salle 2036, Sophie Germain
- **Trois devoirs à la maison** distribués pendant le cours et à rendre pendant le TD :
  - ▶ DM 1: distribué la semaine du 16/02 et à rendre la semaine du 23/02
  - ▶ DM 2: distribué la semaine du 09/03 et à rendre la semaine du 16/03
  - ▶ DM 3: distribué la semaine du 30/03 et à rendre la semaine du 06/04
- **Les devoirs à la maison sont notés.**

# Modalités du cours

- Examen partiel **obligatoire** : mi-mars.

- Note 1ère session :

$$DM = (DM1 + DM2 + DM3)/3$$

Si  $DM \geq 10$ , alors Note =  $\frac{1}{2}$  note partiel +  $\frac{1}{2}$  examen final,  
sinon Note =  $\min(\frac{1}{4} DM + \frac{3}{8}$  note partiel +  $\frac{3}{8}$  note examen,  $\frac{1}{2}$  note partiel +  $\frac{1}{2}$  examen final)

- Note session rattrapage :

$$\text{Max}(\text{exam rattrapage}, \frac{1}{2} \text{ note partiel} + \frac{1}{2} \text{ exam rattrapage})$$

- 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.

- Transparents du cours

<http://www.liafa.univ-paris-diderot.fr/~haberm/cours/logique>

**AVERTISSEMENT: LES  
TRANSPARENTS NE  
CONTIENNENT PAS TOUT !**

- Tableau (exemples et démonstrations)

# Bibliographie

- **Logique pour l'info. : introduction à la déduction automatique.**  
*S. Cerrito, VUIBERT.*
- **Mathématiques pour l'informatique.**  
*A. Arnold et I. Guessarian, MASSON.*
- **Logique et fondements de l'informatique.**  
*R. Lassaigne et M. Rougemont, HERMES.*
- **Logic for Computer Scientists**  
*U. Schöning, BIRKHAUSER*
- **Logic for Computer Science.**  
*J. Gallier, WILEY.* Disponible en ligne:  
<http://www.cis.upenn.edu/~jean/gbooks/logic.html>
- **Logicomics.**  
*A. Doxiadis, C. Papadimitriou, A. Papadatos, A. Di Donna, VUIBERT.*

# 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 :
  - ▶ Dédution naturelle.
  - ▶ Gentzen.
  - ▶ Résolution.
  - ▶ Correction et complétude.
- ③ **Calcul des prédicats** :
  - ▶ Syntaxe, sémantique.
  - ▶ Unification et résolution.
  - ▶ ....

---

# Notions préliminaires

---

**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*)

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

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

## 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  $\mathcal{A}_1 \times \dots \times \mathcal{A}_n$ .
- Un ensemble  $A$  est appelé **dénombrable** si et seulement si il existe une fonction injective (voir définition plus tard)  $f$  de  $\mathbf{N}$  vers  $A$ .

# Relations

**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 **binaire**.

- $R$  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  $(x, z) \in R$ .

# Notations

- $(x, y) \in R$  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  $x \leq y$ .

## Exemples

**Exemple :** La relation  $\leq$  (ou  $\geq$ ) sur les entiers naturels est réflexive, la relation  $<$  (ou  $>$ ) sur les entiers naturels est irreflexive.

**Exemple :** La relation  $=$  sur les ensembles est symétrique, la relation  $\leq$  (ou  $\geq$ ) sur les entiers naturels est anti-symétrique.

**Exemple :** La relation  $\subseteq$  (ou  $\supseteq$ ) sur les ensembles est transitive.

# Équivalence

## Définition :

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

**Exercice :** Montrer que

$\sim = \{(x, y) \mid x, y \in \mathbf{N} \text{ et } x - y \text{ est divisible par } 3\}$  est une équivalence.

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

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

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

**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,  $R^* = A \times A$ .

# 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  $f(\mathcal{C}) = \{y \in \mathcal{B} \mid \text{il existe } x \in \mathcal{C} \text{ t.q. } f(x) = y\}$ .

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  
 $Dom(f) = \{x \in \mathcal{A} \mid \text{il existe } y \in \mathcal{B} \text{ t.q. } (x, y) \in f\}$
- L'**image** de  $f$  est  $Im(f) = \{y \in \mathcal{B} \mid \text{il existe } x \in \mathcal{A} \text{ t.q. } (x, y) \in f\}$
- 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 :**  $f(x) = x + 2$ ,  $f^0(x) = x$ ,  $f^1(x) = x + 2$ ,  $f^2(x) = x + 4$ ,  
 $f^3(x) = x + 6$ ,  $\dots$ ,  $f^n(x) = x + 2 * n$ .

**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 ( $\geq 0$ ) est surjective.  $f(x) = x + 2$  sur les entiers naturels n'est pas surjective.

**Définition :** Une fonction est **bijjective** 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.

## 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 :**  $\leq$  or  $\geq$

**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 :**  $<$  ou  $>$

**Exemple :**  $<$  ou  $>$  sur les entiers,  $\subset$  ou  $\supset$  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 *finis* est bien fondé.

---

# Définitions Inductives et preuves par induction

---

# Définitions inductives en informatique

- Syntaxe concrete
- Syntaxe abstraite
- Règles de typage
- Règles d'évaluation
- etc.

# 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 unaire)

Les entiers naturels

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

# Exemples

## Les arbres binaires (règle binaire)

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

## Les mots sur un alphabet $A$

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

## Exemple (plusieurs axiomes, règles unaires et binaires)

Les expressions de la logique propositionnelle sur l'alphabet  $A$

$$\frac{p \in A}{p \text{ expr}}$$

$$\frac{A_1 \text{ expr} \quad A_2 \text{ expr}}{A_1 \vee A_2 \text{ expr}}$$

$$\frac{A_1 \text{ expr} \quad A_2 \text{ expr}}{A_1 \wedge A_2 \text{ expr}}$$

$$\frac{A_1 \text{ expr} \quad A_2 \text{ expr}}{A_1 \rightarrow A_2 \text{ expr}}$$

$$\frac{A \text{ expr}}{\neg A \text{ expr}}$$

## Exemple (plusieurs assertions)

Les forêts de type T

$$\frac{\overline{a\text{vide} \in \text{arbre T}}}{\frac{t \in T \quad f \in \text{foret T}}{\overline{\text{node}(t, f) \in \text{arbre T}}}}$$

$$\frac{\overline{f\text{vide} \in \text{foret T}}}{\frac{A \in \text{arbre T} \quad f \in \text{foret T}}{\overline{\text{add}(A, f) \in \text{foret T}}}}$$

# Dérivation d'une assertion

Une assertion  $A$  est **dérivable** ssi

- $A$  est un axiome

$$\frac{}{A}$$

- ou il y a une règle de la forme

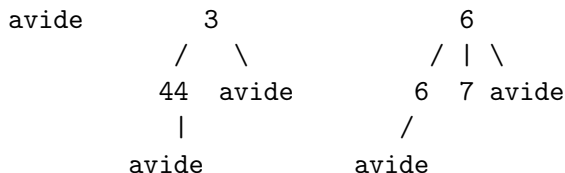
$$\frac{A_1 \quad \dots \quad A_n}{A}$$

telle que  $A_1, \dots, A_n$  sont dérivables

# Dérivation d'une assertion

## Exercice :

- Montrer que  $\text{succ}(\text{succ}(0))$  est naturel est dérivable.
- Donner le terme qui dénote la forêt suivante et montrer comment la construire avec les règles précédentes:



# Ensemble inductif

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

# Preuves par induction (récurrence)

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



# Induction sur les entiers I (induction mathématique)

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

(IM1)  $P(0)$ ,

(IM2) pour tout  $n \in \mathbf{N}$  on a  $P(n)$  implique  $P(n + 1)$

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

alors pour tout  $n \in \mathbf{N}$  on a  $P(n)$  (formellement  $\forall n \in \mathbf{N}. P(n)$ )

## Exemples

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

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”.

## Induction sur les entiers II (induction complète)

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

(IC1)  $P(0)$ ,

(IC2) pour tout  $n \in \mathbf{Nat}$  on a que (pour tout  $k \in \mathbf{Nat}$  t.q.  $k < n$   
on a  $P(k)$ ) implique  $P(n)$

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

alors pour tout  $n \in \mathbf{N}$  on a  $P(n)$  (formellement  $\forall n \in \mathbf{N}. P(n)$ )

# Équivalence des deux principes

Malgré l'apparente supériorité du deuxième principe, on prouve

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

# Le théorème fondamental de l'unité nationale

**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

Soient donnés un ensemble  $\mathcal{A}$ , un ordre strict  $>$  et une propriété  $P$  sur  $\mathcal{A}$ . Un élément **minimal** de  $\mathcal{A}$  est un élément qui n'a pas d'élément plus petit que lui dans  $\mathcal{A}$ .

**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  $P(x)$ ”

## 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.

# Exemples

- **Les mots** : On définit la fonction *concat* comme:  
 $concat(\epsilon, k) = k$  pour tout mot  $k$  et  
 $concat(a.l, k) = a.concat(l, k)$  pour tous mots  $k$  et  $l$  et lettre  $a$   
 $P(m)$  est la propriété :  
 $concat(concat(m, v_1), v_2) = concat(m, concat(v_1, v_2))$
- **Les arbres binaires** :  
 $P(a)$  est la propriété :  $feuilles(a) = noeuds\_internes(a) + 1$



# 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:

$$(x_1, \dots, x_n) >_{lex} (x'_1, \dots, x'_n) \text{ ssi } \begin{array}{l} \text{il existe un } j \text{ avec } 1 \leq j \leq n \text{ tel que} \\ (x_j >_{A_j} x'_j \text{ et pour tout } i \text{ avec } 1 \leq i < j \\ x_i = x'_i) \end{array}$$

**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.

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

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

## Ordres multi-ensembles

**Définition :**  $\mathcal{M} >_{mul} \mathcal{N}$  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  $\mathcal{A}$ .

## Exemple

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
- 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  $\neg$ .

<b>Exemple :</b>	$\neg(p)$	$\vee(p, p)$	$\rightarrow (\wedge(p, q), \neg(r))$
<b>Autre notation :</b>	$\neg p$	$p \vee p$	$(p \wedge q) \rightarrow \neg r$

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

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

sous-formules, etc.

L'ensemble  $\mathcal{SF}(A)$  des sous-formules d'une formule  $A$  est défini inductivement:

- 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  $B\#C$ ,  $\mathcal{SF}(A) = \{B\#C\} \cup \mathcal{SF}(B) \cup \mathcal{SF}(C)$ .

Le nombre d'opérateurs  $op(A)$  d'une formule  $A$  est définie inductivement:

- Si  $A$  est une lettre  $p$ ,  $op(A) = 0$ .
- Si  $A$  est  $\neg B$ ,  $op(A) = op(B) + 1$ .
- Si  $A$  est  $B\#C$ ,  $op(A) = op(B) + op(C) + 1$ .

**Théorème :** Pour toute formule  $A$  on a  $|\mathcal{SF}(A)| \leq 2 * op(A) + 1$ .

Preuve au tableau.

# 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  $A$ .

# La fonction booléenne unaire

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

# Les fonctions booléennes binaires

$$\mathcal{FB}_\vee(\mathbf{V}, \mathbf{V}) = \mathbf{V}$$

$$\mathcal{FB}_\vee(\mathbf{V}, \mathbf{F}) = \mathbf{V}$$

$$\mathcal{FB}_\vee(\mathbf{F}, \mathbf{V}) = \mathbf{V}$$

$$\mathcal{FB}_\vee(\mathbf{F}, \mathbf{F}) = \mathbf{F}$$

$$\mathcal{FB}_\wedge(\mathbf{V}, \mathbf{V}) = \mathbf{V}$$

$$\mathcal{FB}_\wedge(\mathbf{V}, \mathbf{F}) = \mathbf{F}$$

$$\mathcal{FB}_\wedge(\mathbf{F}, \mathbf{V}) = \mathbf{F}$$

$$\mathcal{FB}_\wedge(\mathbf{F}, \mathbf{F}) = \mathbf{F}$$

$$\mathcal{FB}_\rightarrow(\mathbf{V}, \mathbf{V}) = \mathbf{V}$$

$$\mathcal{FB}_\rightarrow(\mathbf{V}, \mathbf{F}) = \mathbf{F}$$

$$\mathcal{FB}_\rightarrow(\mathbf{F}, \mathbf{V}) = \mathbf{V}$$

$$\mathcal{FB}_\rightarrow(\mathbf{F}, \mathbf{F}) = \mathbf{V}$$

## Valeur de vérité d'une formule $A$ par rapport à une interprétation $I$

- 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  $[A]_I = \mathbf{F}$ .

# Formules satisfaisables, contradictoires, valides

**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).

# Conséquence logique et validité

**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  $A$ .

**Lemme :**(Substitution et Validité)

Soit  $A$  une formule et soit  $p$  une de ses lettres propositionnelles. Soit  $A'$  la formule obtenue à partir de  $A$  en remplaçant systématiquement  $p$  par une formule quelconque  $B$ . Si  $A$  est valide, alors  $A'$  est valide aussi.

## 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.

**Lemme :** (Remplacement équivalent)

Soit  $A, B, C$  trois formules et  $B$  une sous-formule de  $A$ . Si  $B \equiv C$  alors  $A \equiv A'$  où  $A'$  est obtenu à partir de  $A$  en remplaçant  $B$  par  $C$ .

## 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**.