# Logique

Pascal Brisset

ENAC

# Logique aristotélicienne (Aristote & Platon)

Proposition : nom + verbe, discours dans lequel réside le vrai ou le faux.
Étude systématique : 4 types de propositions.

Affirmative univ. ($\forall$)

*Tous les hommes sont mortels*

Négative univ. ($\neg\forall$)

*Aucun homme n'est noir*


Aff. particulière($\exists$)

*Certains hommes sont roux*

Nég. part. ($\neg\exists$)

*Certains ne sont pas blancs*

# Syllogisme

Tous les mammifères sont des vertébrés (*prémisse majeure*)
Tous les hommes sont des mammifères (*prémisse mineure*)
Donc tous les hommes sont des vertébrés (*conclusion*)

Étude systématique des syllogismes valides ($4^3 \times 2 \times 2 = 256$ syllogismes)

# Logique stoïcienne (Philon & Mégane)

| | |
|---|---|
| Tous les hommes sont mortels | Tous les $A$ sont $B$ |
| Tous les non-mortels sont non hommes | Tous les non $A$ sont non $B$ |
| S'il fait jour il fait clair | Si $p$ alors $q$ |
| Donc s'il ne fait pas clair, il ne fait pas jour | Donc si non $q$ alors non $p$ |

Notion de *schéma* et d'*instance* : variables propositionnelles. Les stoïciens reconnaissent 5 shémas (dont le syllogisme)

# Leibnitz, Boole ...

Formalisation :

- élimination de la langue naturelle au profit d'un langage ad hoc ;
- calcul sans interprétation ;
- analogie avec l'algèbre classique ;
- notion de preuve : on peut *prouver* un syllogisme.

# Trois étapes

- Langage de formules : suite de caractères
- Interprétation des formules : vraies ou fausses
- Démonstration : calcul et propriétés

## Définition

Soit $\mathcal{V}_p = \{p, q, r, \ldots\}$ un ensemble infini de variables propositionnelles ou atomes.

Ces atomes désignent les énoncés vrais ou faux.

## Définition

Connecteurs : $\leftrightarrow$ (équivalent), $\rightarrow$ (implique), $\wedge$ (et), $\vee$ (ou), $\neg$ (non)

## Définition

Formules : L'ensemble des formules $\mathcal{F}$ est la fermeture transitive de $\mathcal{V}_p$ par les connecteurs :

- Les atomes sont des formules $(\mathcal{V}_p \subset F)$ ;
- Si $A$ et $B$ sont des formules alors $A \wedge B$ est une formule ;
- ...
- Si $A$ est une formule alors $\neg A$ est une formule.

## Propriété

Soit $F_0 \stackrel{\text{def}}{=} \mathcal{V}_p$ et $F_{n+1} \stackrel{\text{def}}{=} F_n \cup \{\neg A / A \in F_n\} \cup \{A \vee B / A, B \in F_n\} \cup \ldots$.
Alors $\cup_{n \in N} F_n = F$

## Définition

On appelle littéral (positif ou négatif) un atome ou sa négation ($p$, $\neg q$, ...).

Exemple : $p$, $p \wedge q$, $(p \vee \neg p) \wedge q$, $(p \rightarrow q) \rightarrow r$ ...

Attention : ne pas confondre variable propositionnelle ($p$, $q$, ...) et *méta-variable* ($A$, $B$, ...)

# Sémantique

## Définition

On appelle valuation ou interprétation une application $\mathcal{I}$ de $\mathcal{V}_p$ vers $\{v, f\}$

## Propriété

Soit $\mathcal{I}$ une interprétation, $A, B$ des formules. L'application $\overline{\mathcal{I}}$ de $F$ dans $\{v, f\}$ existe :

$$\begin{aligned}
\overline{\mathcal{I}}(A) = \mathcal{I}(A) \quad &\text{si} \quad A \in \mathcal{V}_p \\
\overline{\mathcal{I}}(A \wedge B) = v \quad &\text{ssi} \quad \overline{\mathcal{I}}(A) = v \text{ et } \overline{\mathcal{I}}(B) = v \\
\overline{\mathcal{I}}(A \vee B) = v \quad &\text{ssi} \quad \overline{\mathcal{I}}(A) = v \text{ ou } \overline{\mathcal{I}}(B) = v \\
\overline{\mathcal{I}}(A \rightarrow B) = v \quad &\text{ssi} \quad \overline{\mathcal{I}}(A) = f \text{ ou } \overline{\mathcal{I}}(B) = v \\
\overline{\mathcal{I}}(A \leftrightarrow B) = v \quad &\text{ssi} \quad \overline{\mathcal{I}}(A) = \overline{\mathcal{I}}(B) \\
\overline{\mathcal{I}}(\neg A) = v \quad &\text{ssi} \quad \overline{\mathcal{I}}(A) = f
\end{aligned}$$

## Définition

On note $\overline{\mathcal{I}}(A) = v$ par

$$\mathcal{I} \models A$$

pour "$A$ est satisfiable pour $\mathcal{I}$", "$\mathcal{I}$ satisfait $A$", "$\mathcal{I}$ est un modèle de $A$".

## Définition

Une formule $A$ est satisfiable si il existe une interprétation qui la satisfait

Les interprétations peuvent être représentées par une table de vérité :

| $A$ | $B$ | $A \wedge B$ | $A \rightarrow B$ | $\neg A$ | ... |
|---|---|---|---|---|---|
| v | v | v | v | f | ... |
| f | v | f | v | v | ... |
| v | f | f | f | f | ... |
| f | f | f | v | v | ... |

Remarque : l'implication est *classique* (non *intuitioniste* ou *constructive*).
Exemple : construire la table pour $(a \rightarrow b) \rightarrow (b \rightarrow c)$

## Définition

Une tautologie est une formule vraie pour toute valuation

$$\models A$$

## Propriété

Soit $A$ une formule, $p_1 \ldots p_n$ les variables apparaissant dans $A$. Soient $A_1 \ldots A_n$ des formules et $A* \stackrel{\text{def}}{=} A[p_1 \leftarrow A_1] \ldots [p_n \leftarrow A_n]$ une instance de $A$. Si $\models A$ alors $\models A*$.

Exemple : $\models p \rightarrow p$ donc $\models (p \rightarrow q) \rightarrow (p \rightarrow q)$.
**Attention** : la réciproque est fausse.

## Définition

$A$ est équivalent à $B$, $A \equiv B$, ssi pour toute valuation $\mathcal{I}$, $\mathcal{I}(A) = \mathcal{I}(B)$.

## Propriété

$A \equiv B$ si et seulement si $\models A \leftrightarrow B$.

Quelques équivalences de base :

$$
\begin{array}{rcll}
A \to B & \equiv & \neg A \vee B & \\
p \wedge q & \equiv & \neg(\neg p \vee \neg q) & \text{loi de Morgan} \\
p \wedge q & \equiv & q \wedge p & \text{commutativité} \\
p \wedge p & \equiv & p & \text{idempotence} \\
p \wedge (q \vee r) & \equiv & (p \wedge q) \vee (p \wedge r) & \text{distributivité} \\
p \wedge (q \wedge r) & \equiv & (p \wedge q) \wedge r & \text{associativité}
\end{array}
$$

**Propriété**

$\{\wedge, \neg\}$ est un ensemble complet de connecteurs, i.e. pour toute formule $A$, il existe $A* \equiv A$ telle que $A*$ ne s'écrit qu'avec ces deux connecteurs.

Exemple : transformer $(p \to q) \to ((p \wedge q) \leftrightarrow p)$

## Définition

Soient $A_1 \ldots A_n$, $B$ des formules. $B$ est conséquence de $A_1 \ldots A_n$

$$A_1 \ldots A_n \models B$$

ssi pour toute valuation $\mathcal{I}$, si $\mathcal{I}(A_1) = \cdots = \mathcal{I}(A_n) = v$ alors $\mathcal{I}(B) = v$.

Remarque : $A_1, \ldots A_n \models B$ ssi $A_1 \wedge \ldots \wedge A_n \models B$

## Propriété

$A \models B$ ssi $\models A \rightarrow B$

**Preuve** : $\Rightarrow$ :$\forall \mathcal{I}$, si $\mathcal{I}(A) = v$ alors $\mathcal{I}(B) = v$ (car $A \models B$) donc $\mathcal{I}(A \rightarrow B) = v$, si $\mathcal{I}(A) = f$ alors $\mathcal{I}(A \rightarrow B) = v$.
$\Leftarrow$ : $\forall \mathcal{I}$, $\mathcal{I}(A \rightarrow B) = v$ donc si $\mathcal{I}(A) = v$ alors $I(B) = v$ par définition de $\mathcal{I}(\rightarrow)$.

## Propriété

Réfutation : $A_1, \ldots, A_n \models A$ ssi non ($\models A_1 \wedge \ldots \wedge A_n \wedge \neg A$).

Considérons l'ensemble d'atomes fini : $\mathcal{V}_p = \{p_1, \ldots, p_n\}$.

- une formule est une fonction de $\mathcal{V}_p$ dans $\{v, f\}$ ;
- deux formules équivalentes représentent la même fonction.

Donc $card(F/\equiv) \leq 2^{2^n}$. En fait $card(F/\equiv) = 2^{2^n}$ : soit une fonction $f$ donnée

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $p_1$ | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| $p_2$ | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| $p_3$ | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| $f$ | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |

On peut construire la formule $F$ représentant $f$ :

$$F = (p_1 \wedge \neg p_2 \wedge \neg p_3) \vee (\neg p_1 \wedge p_2 \wedge \neg p_3) \vee (p_1 \wedge \neg p_2 \wedge p_3) \ldots$$

## Définition

Forme normale conjonctive (resp. disjonctive) : conjonction (resp. disjonction) de disjonctions (resp. conjonctions) de littéraux.

## Propriété

Toute formule est logiquement équivalente à une formule sous forme normale.

Algorithme de mise sous forme normale ;

1. 
   - $A \leftrightarrow B \equiv (A \rightarrow B) \wedge (B \rightarrow A)$
   - $A \rightarrow B \equiv \neg A \vee B$
2. 
   - $\neg\neg A \equiv A$
   - $\neg(A \vee B) \equiv \neg A \wedge \neg B$
   - $\neg(A \wedge B) \equiv \neg A \vee \neg B$
3. 
   - $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
   - $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$

Exemple : $(A \wedge (A \rightarrow C)) \rightarrow C \equiv \ldots$

# Démonstration = Calcul

$A \models B$ : $B$ est vrai si $A$
$A \vdash B$ : $B$ est prouvable à partir de $A$

Logique = "$A \models B$ ssi $A \vdash B$"

| Syntaxe | Sémantique |
|---|---|
| Formule | Fonction |
| Démonstration | Modèle |
| Axiomes, règles d'inférence | Table de vérité |
| Théorème | Tautologie |
| Déduction | Conséquence valide |
| $\vdash$ | $\models$ |

## Définition

Une démonstration est une suite de formules $A_1 \ldots A_n$ telle que :

- soit $A_i$ est l'instance d'un axiome ;
- soit $A_i$ est le résultat de l'application d'une règle d'inférence sur $A_{i_1} \ldots A_{i_k}$ avec $i_i < i \ldots i_k < i$.

| Axiomes | Règles |
|---|---|
| $A \rightarrow (B \rightarrow A)$ | modus ponens |
| $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$ | $\frac{\vdash A \quad \vdash A \rightarrow B}{\vdash B}$ |
| $(\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B)$ | |

Exemple : preuve de $p \rightarrow p$.

## Définition

Déduction : $A_1 \dots A_n \vdash B$ : en ajoutant les axiomes $A_1 \dots A_n$, $B$ est un théorème ("on peut prouver $B$ avec les hypothèses $A_1 \dots A_n$").

## Propriété

Théorème de déduction : $A \vdash B$ ssi $\vdash A \to B$.

**Preuve** : $\Leftarrow$ : une application de MP. $\Rightarrow$ : théorème de Herbrand, preuve par induction sur la hauteur $h$ de l'arbre de preuve :

- Si $h = 0$. nécessairement $A = B$, or $\vdash A \to A$
- $h > 0$, l'arbre de $A \vdash B$ a la forme suivante :

$$\frac{A \vdash X \quad A \vdash X \to B}{A \vdash B}$$

Par induction, on a $\vdash A \to X$ et $\vdash A \to (X \to B)$ D'où

$$\frac{\vdash A \to X \quad \dfrac{\vdash A \to (X \to B) \quad \vdash a \to (b \to c) \to ((a \to b) \to (a \to c))}{\vdash (A \to X) \to (A \to B)}}{\vdash A \to B}$$

## Définition

Une théorie de la preuve ($\vdash$) est dire correcte si tout théorème $A$ ($\vdash A$) est une formule valide ($\models A$).

## Propriété

La méthode axiomatique est correcte pour le calcul propositionnel.

**Preuve** :

- les axiomes sont des tautologies ;
- la règle du Modus Ponens est correcte (exercice).

## Définition

Une théorie de la preuve est dite consistante s'il n'existe pas de formule $A$ telle que $\vdash A$ et $\vdash \neg A$.

## Propriété

La méthode axiomatique est consistante.

## Définition

On dit qu'un théorie de la preuve ($\vdash$) est complète ssi pour toute formule $A$, si $\models A$ alors $\vdash A$.

## Propriété

La méthode axiomatique est complète. On dit aussi que le calcul propositionnel est complet.

**Preuve** : Avec une autre méthode de démonstration...

## Définition

Informelle : une théorie est décidable s'il existe un algorithme qui étant donnée une formule $A$ décide si $\vdash A$ ou si non $\vdash A$.

## Propriété

Le calcul des propositions est décidable.

**Preuve** : Exercice

# Calcul des séquents

Idée : pour montrer qu'une formule est une tautologie, on montre qu'aucune valuation ne satisfait sa négation. Pour cela on manipule deux ensembles de formules : celles qui doivent être fausses (à droite) et celles qui doivent être vraies (à gauche).

Exemple : $(A \to B) \to (\neg B \to \neg A)$

## Définition

Un séquent est une paire d'ensembles finis de formules : $\Gamma \longrightarrow \Delta$, $\Gamma$ est l'antécédent, $\Delta$ le conséquent.

Intuitivement, $A_1 \ldots A_n \longrightarrow B_1 \ldots B_m$ s'interprète comme la formule $A_1 \wedge \ldots A_n \to B_1 \vee \ldots B_m$, i.e. un séquent est faux si tous les $B_i$ sont faux et tous les $A_i$ sont vrais.

# Système G de Gentzen

## Définition

Une règle d'inférence est constituée d'un ensemble de séquents prémisses et d'un séquent conclusion

$$\frac{\textit{prémisses}}{\textit{conclusion}}$$

On va construire un arbre dont les nœuds sont étiquetés par ces règles. $A$ et $B$ sont des formules, $\Gamma$ et $\Delta$ sont des ensembles de formules) :

$$\frac{A, B, \Gamma \longrightarrow \Delta}{A \wedge B, \Gamma \longrightarrow \Delta} \wedge_G \qquad \frac{\Gamma \longrightarrow A, \Delta \quad \Gamma \longrightarrow B, \Delta}{\Gamma \longrightarrow A \wedge B, \Delta} \wedge_D$$

$$\frac{A, \Gamma \longrightarrow \Delta \quad B, \Gamma \longrightarrow \Delta}{A \vee B, \Gamma \longrightarrow \Delta} \vee_G \qquad \frac{\Gamma \longrightarrow A, B, \Delta}{\Gamma \longrightarrow A \vee B, \Delta} \vee_D$$

$$\frac{\Gamma \longrightarrow A, \Delta \quad B, \Gamma \longrightarrow \Delta}{A \to B, \Gamma \longrightarrow \Delta} \to_G \qquad \frac{A, \Gamma \longrightarrow B, \Delta}{\Gamma \longrightarrow A \to B, \Delta} \to_D$$

$$\frac{\Gamma \longrightarrow A, \Delta}{\neg A, \Gamma \longrightarrow \Delta} \neg_G \qquad \frac{A, \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \neg A, \Delta} \neg_D$$

## Définition

On dit que le séquent $A_1 \dots A_n \longrightarrow B_1 \dots B_m$ est falsifiable ssi il existe une valuation $\mathcal{I}$ telle que $\mathcal{I} \models A_1 \wedge \dots \wedge A_n \wedge \neg B_1 \wedge \dots \neg B_m$
On dit que le séquent $A_1 \dots A_n \longrightarrow B_1 \dots B_m$ est valide ssi
$\models A_1 \wedge \dots A_n \rightarrow B_1 \vee \dots B_m$

## Propriété

Les règles du système G sont correctes : le séquent conclusion est valide ssi les séquents prémisses sont valides.

# Arbre de preuve

## Définition

Un axiome est un séquent $\Gamma \longrightarrow \Delta$ tel que $\Gamma \cap \Delta \neq \emptyset$.

Remarque : un axiome est un séquent valide.

## Définition

Arbre de déduction : tous les nœuds sont des règles.
Arbre de preuve : arbre de déduction dont les feuilles sont des axiomes.

Exemple : $\longrightarrow ((A \to B) \wedge A) \to B$

Exercice : Donner la règle d'inférence pour le connecteur ou-exclusif $\oplus = \lambda pq.((p \wedge \neg q) \vee (\neg p \wedge q))$

## Définition

On dit qu'un séquent est prouvable si on peut construire un arbre de preuve dont il est la racine.

## Propriété

Le système G est correct (sain), i.e. un séquent prouvable est valide.

**Preuve** : par induction sur l'arbre de preuve.
En pratique :

- Pour montrer $\models A$, on prouve $\models \longrightarrow A$ ;
- Pour montrer $A \models B$, on prouve $\models \longrightarrow A \rightarrow B$.

# Recherche de preuve et complétude

Algorithme de recherche d'un arbre de preuve $\mathcal{T}$ pour un séquent $\Gamma \longrightarrow \Delta$ :

- Initialisation : $\mathcal{T} := \Gamma \longrightarrow \Delta$
- Choisir dans $\mathcal{T}$ une feuille qui contient au moins une formule non atomique, lui appliquer une règle, recommencer.
- Si le choix est impossible :
  - si toutes les feuilles sont des axiomes, le séquent initial est valide
  - sinon, le séquent initial est falsifiable

Exemple : $\longrightarrow (p \vee (q \rightarrow p)) \rightarrow (p \vee \neg q)$, $\longrightarrow (p \rightarrow q) \rightarrow (q \rightarrow p)$.

## Propriété

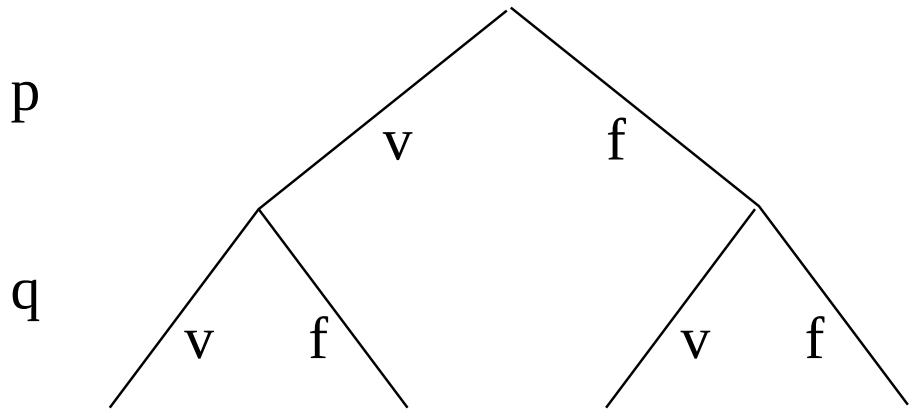
La procédure de recherche termine. Si le séquent initial est valide alors l'arbre obtenu est un arbre de preuve. S'il est falsifiable, on obtient un contre-exemple en falsifiant la feuille qui n'est pas un axiome.

## Corollaire : le système G est complet

# Arbre

Ce n'est pas à proprement parler de la démonstration mais plutôt de l'évaluation.

Méthode identique aux tables de vérités mais sous forme d'arbre : chaque niveau de l'arbre correspond à une variable.
Exemple : $((p \rightarrow q) \wedge q) \rightarrow q$



# Algorithme de QUINE

On évalue partiellement la formule au fur et à mesure de la création de l'arbre.

$$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$$

A: v f

$$(B \to C) \to (B \to C) \qquad v \to (v \to v)$$

B:

Rappel : Toute formule admet une forme normale conjonctive.

## Définition

On appelle clause une disjonction de littéraux. On notera une formule sous forme normale conjonctive comme un ensemble de clauses.

# Algorithme de Davis & Putnam

Méthode analogue à Quine sur la forme normale conjonctive.
Calculer la forme normale conjonctive $N$ de la formule à prouver.

1. Si $N$ est vide, succès.
2. Si une clause de $N$ est vide, la formule n'est pas satisfiable
3. Choisir une variable $p$, calculer $N_p$ (resp. $N_{\neg p}$) l'ensemble des clauses de $N$ où $p$ (resp. $\neg p$) apparaît, $N_c$ le reste.
4. Calculer $\overline{N_p}$ (resp. $\overline{N_{\neg p}}$) en supprimant $p$ dans $N_p$ ($\neg p$ dans $N_{\neg p}$).
5. Répéter récursivement avec $\overline{N_p} \cup N_c$ et $\overline{N_{\neg p}} \cup N_c$

**Preuve** : $N$ est satisfiable ssi $\overline{N_p} \cup N_c$ ou $\overline{N_{\neg p}} \cup N_c$ le sont.
Heuristiques :

- Si une clause ne contient qu'un seul littéral, le choisir.
- Choisir la variable qui a le plus d'occurrences.

Rappel : $A_1, \ldots, A_n \models A$ ssi $A_1 \wedge \ldots \wedge A_n \wedge \neg A$ est insatisfiable.
Donc toute preuve de déduction peur se faire par preuve d'insatisfiabilité : principe de réfutation.

## Propriété

Soient $C_1$ et $C_2$ deux clauses et $p$ un atome tel que $p \in C_1$ et $\neg p \in C_2$.
On appelle $R = C_1 \backslash p \cup C_2 \backslash \neg p$ la résolvante de $C_1$ et $C_2$.

$$\{R, C_1, C_2\} \equiv \{C_1, C_2\}$$

C'est le principe de résolution.

**Preuve** : $(A \vee p) \wedge (B \vee \neg p) \models A \vee B$ (exercice)

## Propriété

La résolution est complète pour la réfutation : en partant d'un ensemble de clauses insatisfiable, on obtient la clause vide par application répétée du principe de résolution.

Exemple : $\{p \rightarrow r, q \rightarrow r\} \models (p \vee q) \rightarrow r$

# Clauses de Horn

## Définition

Une clause de Horn est une clause comportant un littéral positif au plus.

Exemple : $p$, $p \vee \neg q \vee \neg r$, $\neg p \vee \neg q$, $p \vee q$.
Algorithme de résolution pour les clauses de Horn. Soit $N$ l'ensemble de clauses dont on veut prouver la non satisfiabilité.

1. Si $\emptyset \in N$, $N$ est insatisfiable (succès pour une réfutation)
2. Sinon, choisir $\{p\} \in N$ et $C \in N$ tel que $\neg p \in C$, calculer $C \backslash \{\neg p\}$ la résolvante de $\{p\}$ et $C$, continuer avec $N \backslash C \cup \{R\}$
3. Sinon $N$ est satisfiable (échec pour une réfutation)

Remarque : on peut supprimer $C$ de $N$ car $C$ est une conséquence logique de $R$.
Exemple : $\{\neg p \vee r, \neg r \vee s, p, \neg s\}$

# Calcul des propositions paramétré

« Un homme est mortel », « Socrate est mortel » . . .« $x$ est mortel »
On utilise le prédicat « est mortel » appliqué à $x$ : $p(x)$

« Tous sont mortels » : $\forall x\ p(x)$
Socrate (constante $s$) est un terme auquel on peut appliquer $p$ : $p(s)$

Même plan que pour le calcul des propositions :

- langage ;
- interprétation, $\models$ ;
- démonstration, $\vdash$.

- $V = \{x, y, z, \ldots\}$ un ensemble de variables ;
- $\Sigma = \{a, b, f, g \ldots\}$, la signature, un ensemble de symboles fonctionnels, chacun muni d'une arité $\in N$ ; on note $\Sigma_n$ les symboles d'arité $n$ ;
- $P = \{p, q, \ldots\}$, les symboles de prédicats, chacun muni d'une arité $\in N$ ; $P_n$ les symboles d'arité $n$ ;
- $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\neg$ des connecteurs ;
- $\forall$ et $\exists$ des quantificateurs.

**Définition**

[terme] L'ensemble des termes $T$ est le plus petit ensemble contenant $V \cup \Sigma_0$ et clos par : si $f \in \Sigma_n$, $t_1, \ldots, t_n$ des termes alors $f(t_1, \ldots, t_n)$ est un terme.

En particulier, un symbole fonctionnel d'arité 0 est une constante.
Exemple : avec $\Sigma = \{z_0, s_1\}$, on peut former les termes $z$, $s(z)$, $s(s(z))$ ...

## Définition

[atome] Pour tout $p \in P_n$, quel que soient $t_1 \dots t_n$ des termes, $p(t_1, \dots, t_n)$ est un atome

En particulier, un symbole de prédicat d'arité 0 est une proposition.

## Définition

[formule] L'ensemble des formules $F$ est le plus petit ensemble contenant l'ensemble des atomes et vérifiant :

- Si $A$ et $B$ sont des formules alors $A \wedge B$, $A \vee B$, $A \rightarrow B$ et $A \leftrightarrow B$ sont des formules ;
- Si $A$ est une formule alors $\neg A$ est une formule ;
- Si $A$ est une formule et $x$ une variable alors $\forall x A$ et $\exists x A$ sont des formules.

Exemple : la continuité peut s'exprimer en calcul des prédicats

$$\forall x \forall \epsilon \exists \eta \forall y (p(x, y, \eta) \rightarrow q(x, y, \epsilon))$$

avec $p(x, y, \eta) \stackrel{\text{def}}{=} |x - y| < \eta$ et $q(x, y, \epsilon) \stackrel{\text{def}}{=} |f(x) - f(y)| < \epsilon$

## Définition

Une variable libre est une variable non quantifiée, une variable liée est une variable non libre (cf $\lambda$-calcul). Un terme (resp. une formule) clos est un terme (resp. une formule) sans variable libre. On note $V(t)$ les variables libres d'un terme $t$.

On parlera aussi de portée d'un quantificateur :

$$\forall x \overbrace{(p(x) \rightarrow \forall y \underbrace{\exists x \quad \underbrace{q(x,y)}_{\text{portée de } \exists x}}_{\text{portée de } \forall y})}^{\text{portée de } \forall x}$$

$$F \longrightarrow \{0,1\}$$
$$T \longrightarrow ???$$

## Définition

Soit $D$ le domaine des termes, une interprétation $\mathcal{I}$ est une application sur $\Sigma \cup P$ :

$$\Sigma \longrightarrow \bigcup_{n \in N} (D^n \longrightarrow D)$$
$$c \longmapsto \overline{c} \in D \quad \text{constante}$$
$$f_i \longmapsto \overline{f_i} : D^i \longrightarrow D \quad \text{fonction}$$

$$P \longrightarrow \bigcup_{n \in N} (D^n \longrightarrow \{0,1\})$$
$$p_i \longmapsto \overline{p_i} : D^i \longrightarrow \{0,1\} \quad \text{prédicat}$$

## Propriété

Une interprétation peut être étendue aux ensembles de termes et de formules ($\sigma : V \to D$ désigne la valuation des variables libres) :

$$
\begin{aligned}
\mathcal{I}_\sigma(f(t_1,\ldots,t_n)) &\stackrel{\text{def}}{=} \mathcal{I}(f)(\mathcal{I}_\sigma(t_1),\ldots,\mathcal{I}_\sigma(t_n)) \\
\mathcal{I}_\sigma(x) &\stackrel{\text{def}}{=} \sigma(x) \\
\mathcal{I}_\sigma(p(t_1,\ldots,t_n)) &\stackrel{\text{def}}{=} \mathcal{I}(p)(\mathcal{I}_\sigma(t_1),\ldots,\mathcal{I}_\sigma(t_n)) \\
\mathcal{I}_\sigma(A \vee B) &\stackrel{\text{def}}{=} \max(\mathcal{I}_\sigma(A),\mathcal{I}_\sigma(B)) \\
\mathcal{I}_\sigma(A \wedge B) &\stackrel{\text{def}}{=} \min(\mathcal{I}_\sigma(A),\mathcal{I}_\sigma(B)) \\
\mathcal{I}_\sigma(\neg A) &\stackrel{\text{def}}{=} 1 - \mathcal{I}_\sigma(A) \\
\mathcal{I}_\sigma(\forall x A) &\stackrel{\text{def}}{=} \inf_{\sigma'} \mathcal{I}_{\sigma'}(A) \text{ avec } \sigma'(y) = \sigma(y) \text{ pour } y \neq x \\
\mathcal{I}_\sigma(\exists x A) &\stackrel{\text{def}}{=} \sup_{\sigma'} \mathcal{I}_{\sigma'}(A) \text{ avec } \sigma'(y) = \sigma(y) \text{ pour } y \neq x
\end{aligned}
$$

Exemple : soient $\Sigma = \{z_0, u_0, a_2\}$ et $P = \{e_2, p_1\}$. On peut choisir :

$$\begin{array}{rcll} D & = & N & \text{les entiers} \\ \mathcal{I}(z) & = & 0 & \text{zéro} \\ \mathcal{I}(u) & = & 1 & \text{un} \\ \mathcal{I}(a) & = & \lambda xy.(x+y) & \text{l'addition} \\ \mathcal{I}(e) & = & \lambda xy.(x=y) & \text{l'égalité} \\ \mathcal{I}(p) & = & \lambda x.(x=0) & \text{l'égalité à zéro} \end{array}$$

Que signifient alors :

- $\neg e(z,u)$
- $\forall x \forall y\ e(a(x,y),a(y,x))$
- $\exists x\ e(x,a(z,u))$
- $p(e(z,u))$

Remarque : l'ensemble des termes étant en général infini il n'est pas possible de représenter l'interprétation d'un prédicat par une table de vérité.

## Définition

On dit qu'une formule $A$ close est satisfiable si il existe un domaine $D$ et une interprétation $\mathcal{I}$ telle que $\mathcal{I}_\emptyset(A) = 1$. On dit alors que $(D, \mathcal{I})$ est un modèle pour $A$.
Conséquence logique ($A \models B$), équivalence ($A \equiv B$) : mêmes définitions que pour les propositions.

Remarque : en général ce n'est pas l'opération de recherche de modèle que l'on doit faire, mais l'inverse, l'axiomatisation d'un domaine $D$ donné : quel est l'ensemble de formules qui a pour (unique) modèle $D$.

## Exercices

- Trouver des modèles pour les formules suivantes
  - $\forall x\ p(x) \wedge \exists x\ \neg p(x)$
  - $\{\forall x\ e(f(x,a),x) \wedge e(f(a,x),x),\ \forall x \forall y \forall z\ e(f(f(x,y),z),f(x,f(y,z)))\}$
  - $\exists y \forall x\ r(y,x) \leftrightarrow \neg r(x,x)$
- Que dire de « *Ce barbier rase tous les hommes qui ne se rasent pas eux mêmes* » (axiomatiser et (re)-modéliser) ?
- Que doit vérifier la signature $\Sigma$ pour que l'ensemble des termes soit fini ?

La mise sous forme normale va nous permettre de :

- simplifier les formules en supprimant les quantificateurs ;
- pouvoir faire de la démonstration par résolution.

# Forme prénexe

## Définition

Une formule est sous forme prénexe si elle s'écrit

$$\mathcal{Q}_1 x_1 \dots \mathcal{Q}_n x_n \; F$$

où les $\mathcal{Q}_i$ sont des quantificateurs ($\forall$ ou $\exists$) et $F$ est une formule sans quantificateur.

## Propriété

Toute formule est équivalente à une formule sous forme prénexe.

**Preuve** : transformation de la formule :

1. Supprimer les équivalences ($\leftrightarrow$) et les implications ($\rightarrow$) :
   - $A \leftrightarrow B \equiv (A \rightarrow B) \wedge (B \rightarrow A)$
   - $A \rightarrow B \equiv \neg A \vee B$
2. Faire « descendre » les négations :
   - $\neg \forall x\ A \equiv \exists x\ \neg A$
   - $\neg \exists x\ A \equiv \forall x\ \neg A$
3. Faire « remonter » les quantificateurs en renommant les variables ($\alpha$-conversion) et en utilisant l'équivalence

$$(\mathcal{Q}x\ A) \wedge\ B \equiv B \wedge (\mathcal{Q}x\ A) \equiv \mathcal{Q}x\ (A\ \wedge\ B) \qquad \text{(idem pour } \vee)$$

où $\mathcal{Q}$ est un quantificateur et $x$ une variable non libre dans $B$.

Exemple : $\forall x\ p(x) \rightarrow \exists x\ p(x)$

# Skolémisation

Suppression des quantificateurs existentiels ($\exists$).

## Définition

[skolémisation] On obtient la forme skolémisée d'une formule sous forme prénexe en :

- associant à chaque variable $x$ quantifiée existentiellement un (nouveau) symbole fonctionnel $f$ (éventuellement une constante) d'arité égale au nombre de quantifications universelles antérieures $x_1 \dots x_n$ ;
- supprimant la quantification existentielle ;
- remplaçant chaque occurrence de la variable $x$ par le terme $f(x_1, \dots, x_n)$.

Attention : la formule skolémisée n'est pas équivalente à la formule originale.

Exemple : $\forall x(\forall y\ p(x,y) \rightarrow \neg\exists z\ q(x,z))$

## Propriété

Soit $A^*$ la forme skolémisée d'une formule $A$.

1. $A^* \models A$ ;
2. $A$ est satisfiable ssi $A^*$ l'est.

Remarques :

- les quantifications universelles d'une formule prénexe et skolémisée étant inutiles, on les oublie ;
- une formule prénexe et skolémisée peut être mise sous forme normale conjonctive (clausale).

## Définition

Une formule prénexe-skolémisée-clausale est dite sous forme standard.

## Définition

Soit $C$ une clause. $C$ où toutes les variables sont remplacées par un terme clos est appelée instance close de $C$

Exemple : $\Sigma = \{a_0, f_1\}$, $C = \{p(x), q(x, y)\}$ : $\{p(a), q(a, f(a))\}$
On peut alors appliquer la méthode de résolution (propositionnelle) à des instances closes.
Exemple : Montrer que

$$\underbrace{\forall x\ p(x) \rightarrow q(x)}_{H_1},\ \underbrace{\forall x\ q(x) \rightarrow r(x)}_{H_2} \models \underbrace{\forall x\ p(x) \rightarrow r(x)}_{C}$$

On montre que $H_1 \wedge H_2 \wedge \neg C$ n'est pas satisfiable.
Forme prénexe :

$$\exists z \forall x \forall y\ (p(x) \rightarrow q(x)) \wedge (q(y) \rightarrow r(y)) \wedge p(z) \wedge \neg r(z)$$

Skolémisation (nouvelle constante $c$), forme clausale :

$$\{p(x) \rightarrow q(x), q(y) \rightarrow r(y), p(c), \neg r(c)\}$$

Instances closes (avec la signature minimale : $\Sigma = \{c\}$) :

$$\{\underbrace{p(c) \to q(c)}_{\boxed{1}}, \underbrace{q(c) \to r(c)}_{\boxed{2}}, \underbrace{p(c)}_{\boxed{3}}, \underbrace{\neg r(c)}_{\boxed{4}}\}$$

Résolution :

$$\boxed{2} \text{ et } \boxed{4} \quad \mapsto \quad \underbrace{\neg q(c)}_{\boxed{5}}$$

$$\boxed{5} \text{ et } \boxed{1} \quad \mapsto \quad \underbrace{\neg p(c)}_{\boxed{6}}$$

$$\boxed{6} \text{ et } \boxed{3} \quad \mapsto$$

Un théorème de HERBRAND nous assure qu'on peut se restreindre à cette *signature minimale* (l'ensemble des symboles fonctionnels qui apparaissent dans les clauses).

# Unification

## Définition

Une substitution $\sigma$ est une application de l'ensemble des variables $V$ dans l'ensemble des termes. Une substitution peut être étendue à l'ensemble des termes et des atomes.

Exemple : $\Sigma = \{a_0, f_1, \ldots\}$, $P = \{p_1, \ldots\}$. On définit $\sigma$ par

$$\begin{array}{rcl} x & \mapsto & a \\ y & \mapsto & f(z) \end{array}$$

alors

$$\begin{array}{rcll} \sigma(f(x)) & = & f(a) & \text{terme} \\ \sigma(p(f(x))) & = & p(f(a)) & \text{atome} \\ \sigma(p(y)) & = & p(f(z)) & \end{array}$$

On pourra noter $\sigma = \{x \leftarrow a, y \leftarrow f(z)\}$

## Définition

Soit deux termes $t_1$ et $t_2$. On appelle unificateur de $t_1$ et $t_2$ une substitution $\sigma$ telle que $\sigma(t_1) = \sigma(t_2)$ ($t_1$ et $t_2$ sont unifiables).
On appelle $\sigma_u$ l'unificateur le plus général, l'unificateur tel que pour tout unificateur $\sigma'$, pour tout terme $t$, $\sigma'(t)$ est une instance de $\sigma(t)$.

Intuition : $\sigma_u$ est l'unificateur qui « fait le minimum ».
Exemple : pour $< f(x, a), f(y, z) >$

- $\{x \leftarrow a, y \leftarrow a, z \leftarrow a\}$ est un unificateur ;
- $\{x \leftarrow y, z \leftarrow a\}$ est le plus général.

Algorithme d'unification de deux termes $t_1$ et $t_2$

1. Initialisation : $E := \{< t_1, t_2 >\}$, $\sigma := \emptyset$
2. Tant que $E \neq \emptyset$
   1. Soit $E := E \setminus < t, t' >$
   2. Si $t = t'$ alors continuer
   3. Si $t = x \in V$ (resp. pour $t' = x \in V$)
      1. Si $x \in V(t')$ alors échec (test d'occurrence)
      2. Sinon, $\sigma := \sigma \circ \{x \leftarrow t'\}$, $E := \sigma(E)$
   4. Sinon $t = f(t_1, \ldots, t_n)$, $t' = f'(t'_1, \ldots, t'_{n'})$
      1. Si $f \neq f'$ alors échec
      2. Sinon $E := E \cup \{< t_i, t'_i > / 1 \leq i \leq n\}$
3. $\sigma$ est l'unificateur le plus général (s'il n'y a pas eu échec)

Exemple : $\Sigma = \{f_3, a_0, g_1\}$, $x, y, z \in V$

$$< f(a, g(x), x), f(y, g(y), z) >$$

# Principe de résolution

## Définition

Soient deux clauses $C$ et $C'$ telles que $A \in C$, $\neg A' \in C'$ et $A$ et $A'$ sont unifiables par $\sigma$ le plus grand unificateur. $R = \sigma(C \backslash A \cup C' \backslash \neg A')$ est la résolvante de $C$ et $C'$.

## Propriété

$\{C, C'\}$ et $\{C, C', R\}$ sont équivalents

## Propriété

Soit $E$ un ensemble de clauses. $E$ est insatisfiable si et seulement si on peut obtenir la clause vide par application répétée du principe de résolution.

Exemple : Montrer que $s(j)$ est déductible de

$$\forall x \forall y \ (h(y) \wedge i(x,y) \wedge a(x) \rightarrow d(x))$$
$$\forall x \ (\forall y \ (i(y,x) \wedge b(y)) \rightarrow d(y)) \rightarrow s(x)$$
$$h(j)$$
$$\forall x \ (b(x) \rightarrow a(x))$$

Inconvénient de la résolution : perte de structure des formules pour la démonstration.
Le calcul des séquents propositionnel peut être étendu au calcul des prédicats en ajoutant des règles pour traiter les quantificateurs.

$$\frac{\Sigma; p(t), \Gamma \longrightarrow \Delta}{\Sigma; \forall x.p(x) \in \Gamma \longrightarrow \Delta}\forall_G \qquad \frac{\Sigma + c; \Gamma \longrightarrow p(c), \Delta}{\Sigma; \Gamma \longrightarrow \forall x.p(x), \Delta}\forall_D$$

$$\frac{\Sigma + c; p(c), \Gamma \longrightarrow \Delta}{\Sigma; \exists x.p(x), \Gamma \longrightarrow \Delta}\exists_G \qquad \frac{\Sigma; \Gamma \longrightarrow p(t), \Delta}{\Sigma; \Gamma \longrightarrow \exists x.p(x) \in \Delta}\exists_D$$

- dans les règles $\forall_G$ et $\exists_D$, $t$ désigne un terme quelconque sur la signature $\Sigma$ ;
- dans les règles $\forall_D$ et $\exists_G$, $c$ désigne une nouvelle constante qui est ajoutée à la signature pour la suite du calcul.

Pratiquement, dans le cas d'une preuve faite en chaînage arrière, on ne peut pas « deviner » le terme $t$ au moment de l'utilisation de la règle $\forall_G$ ou $\exists_D$ : on laisse une variable qui sera remplacée par le terme lui-même le moment voulu (en respectant la contrainte sur les constantes universelles).

Exemple : $\Sigma = \{f_1, a_0\}$, $\overset{?}{\models} (\forall x\ p(x)) \longrightarrow \forall y\ p(f(y))$

$$\dfrac{\dfrac{\dfrac{\dfrac{t = f(c)}{\Sigma + c; p(t) \longrightarrow p(f(c))}\,axiome}{\Sigma + c; \forall x\ p(x) \longrightarrow p(f(c))}\,\forall_G}{\Sigma; \forall x\ p(x) \longrightarrow \forall y\ p(f(y))}\,\forall_D}{\Sigma; \longrightarrow (\forall x\ p(x)) \rightarrow \forall y\ p(f(y))}\,\longrightarrow_D$$

$t$ est un $\Sigma + c$-terme

$c$ est une nouvelle constante

Exercice :

- Essayer d'appliquer $\forall_G$ **avant** $\forall_D$ dans cette démonstration.
- $\overset{?}{\models} (\forall x\ p(f(x))) \longrightarrow \forall y\ p(y)$

Bibliographie :

- Informatique théorique & intelligence artificielle, Alliot & Schiex
- Symbolic Logic & Mechanical Theorem Proving, Chang & Lee
- Logic in Computer Science, Gallier
- Logique, Gochet & Gribomont