Logique Mathématique

Plan du cours

- Introduction
- Logique des propositions
- Logique du premier ordre
- Logiques non classiques

Plan

- Introduction

Définition

Logique (n.f.)

- Étude scientifique des conditions de vérité des propositions
- Manière de raisonner
- Enchaînement cohérent d'idées

Logique et Informatique

- Le calcul des prédicats est la base de beaucoup de formalismes de représentation et de manipulation de la connaissance en intelligence artificielle.
- Ici, on veut modéliser le raisonnement, c'est-à-dire le processus permettant d'inférer des expressions "logiquement correctes" à partir des faits.
- La capacité de modéliser le raisonnement, si approximatif soit-il, nous permettra de le programmer dans des systèmes informatisés.

Logique et Informatique

- La **programmation logique** est une forme de programmation qui définit les applications à l'aide d'un ensemble de faits élémentaires les concernant et de règles de logique leur associant des conséquences plus ou moins directes.
- Ces faits et ces règles sont exploités par un démonstrateur de théorème ou moteur d'inférence
- **Prolog** est un exemple de langage de programmation logique

Logique et Informatique

- Logique et calculabilité

  Turing Church
- Logique et complexité

Démarche du cours

Étude de la logique binaire dans la perspective de la démonstration automatique.

- Logique des propositions
- Logique du premier ordre

Étude de logiques non classiques

- Quelques extensions de la logique classique
- Quelques logiques rivales

Démarche du cours

• En premier lieu, nous allons considérer la syntaxe, c-à-d., la forme des expressions qu'on peut écrire dans le calcul des prédicats.

• Ensuite, nous considérons la sémantique, c-à-d., comment déterminer si une expression est logiquement correcte.

• Plus tard, nous verrons une méthode de *déduction*, c-à-d, une méthode pour déterminer des expressions qui sont des conséquences logiques des autres.

Le Calcul Propositionnel …

But du calcul propositionnel : donner un fondement formel à un ensemble restreint d'énoncés du langage en utilisant :

- des propositions élémentaires (« Il fait beau », …).
- des connecteurs (et, ou, donc, équivalent à) et
- la négation

Exemple : phrase *p* = « *Il fait beau et je suis en vacances* »

p est vraie si les deux propositions *q* et *r* sont vraies :

q = « *Il fait beau* »

r = « *Je suis en vacances* »

Définitions

- Une proposition est un énoncé du langage ordinaire considéré du point de vue formel. Cet énoncé est soit vrai soit faux mais pas les deux.
- Exemple: « Le chat du voisin est mort » est une proposition

Définitions

- Valeur de vérité:

  Proposition vraie si adéquation entre proposition et faits du monde réel, fausse sinon

Exemple: Il est mardi et Le chat du voisin est mort

Plan

- Introduction
- Logique des propositions
  - **Syntaxe**
- Logique du premier ordre
- Logiques non classiques

Symboles du Calcul Propositionnel …

- Symboles (ou atomes) propositionnels : P, Q, R, S, ... (dernières lettres majuscules de l'alphabet)

  Dénotent des propositions ou des phrases, pouvant être vraies ou fausses, à propos d'un environnement (« la voiture est rouge », « l'eau est froide », …). Il s'agit d'énoncés élémentaires

- Symboles de vérité : vrai, faux

  Connecteurs propositionnels : équivalence ≡ (ou ↔)

  implication →

  conjonction ∧

  disjonction ∨

  négation ¬

Formules du Calcul Propositionnel …

- Les formules (ou formules bien formées FBF) du calcul propositionnel, notées par les lettres majuscules de l'alphabet (A, B, C …), sont définies par :

  1) Les atomes (vrai, P, Q, R, …) sont des formules.

  2) Si A et B sont deux formules, alors $(A \rightarrow B)$, $(A \equiv B)$, $(A \wedge B)$ et $(A \vee B)$ sont des formules.

  3) Si A est une formule, alors $(\neg A)$ est une formule.

- Symboles ( ) et [ ] : utilisés pour grouper des formules.

$$(P \vee Q) \equiv R \quad \neq \quad P \vee (Q \equiv R)$$

Formules du Calcul Propositionnel (suite) …

- Une expression est une FBF si et seulement si elle a été formulée à partir de symboles légaux et des règles utilisant les connecteurs.
- **Exemple : ((P ∧ Q) → R) ≡ ⌉P ∨ ⌉Q ∨ R** est une FBF puisque :

  P, Q, R sont des FBFs,

  P ∧ Q est une FBF (conjonctions de deux FBFs),

  (P ∧ Q) → R est une FBF (implication entre deux FBFs),

  ⌉P et ⌉Q sont deux FBFs (négation de FBFs),

  ⌉P ∨ ⌉Q est une FBF (disjonction de deux FBFs),

  ⌉P ∨ ⌉Q ∨ R est une FBF (disjonction de 3 FBFs) et

  ((P ∧ Q) → R) ≡ ⌉P ∨ ⌉Q ∨ R est une FBF (équivalence entre deux FBFs).

Règles d'élimination des parenthèses

- Supprimer les parenthèses entourant les variables
- Tenir compte de la priorité des connecteurs

ordre standard : $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\equiv$

- Considérer qu'un opérateur unaire l'emporte toujours sur un opérateur binaire
- Exemple:

Backus-Naur-Form

- G = {VN ,<proposition>, R }
- VN = { <proposition>, <implication>, <terme>, <facteur>, <proposition secondaire>, <proposition primaire>}
- R est donné sous la forme BNF suivante

<prop.> ::= <impl.> | <prop.> ≡ <impl.>
<impl.> ::= <terme> | <implication> → <terme>
<terme> ::= <fact.> | <terme> ∨ <fact.>
<fact.> ::= <prop. sec.> | <fact.> ∧ <prop. sec.>
<prop. sec.> ::= <prop. prim.> | ¬ <prop. prim.>
<prop. prim.>::= (<prop.>) | p (avec p € P)

Plan

- Introduction
- Logique des propositions
  - **Syntaxe**
  - **Sémantique**
- Logique du premier ordre
- Logiques non classiques

Sémantique du Calcul Propositionnel …

- **But :** établir un mécanisme **sémantique** d'évaluation des FBFs.

- Interprétation

- donner un sens à nos symboles
- donner la valeur de vérité des atomes

Exemple

Associer l'atome Bat_OK a la proposition P « la batterie est chargée »

La valeur de vérité (vraie, faux) de Bat_OK dépend de la valeur de la proposition P. (qui a son tour dépend, par exemple, d'un senseur X qui mesure le voltage de la batterie)

Sémantique du Calcul Propositionnel …

Symbole → Une valeur de vérité : vrai (V) ou faux (F)

Connecteur → Une table de vérité

- donner la valeur de vérité des connecteurs qui relient ces atomes

P	Q	⌉P	P ∨ Q	P ∧ Q	P → Q	P ≡ Q
V	V	F	V	V	V	V
V	F	F	V	F	F	F
F	V	V	V	F	V	F
F	F	V	F	F	V	V

Exemple: (⌉ **P → Q) ∧ (Q ≡ R)**

Sémantique du Calcul Propositionnel …

• Une interprétation **satisfait** une FBF si la FBF à la valeur **T** pour cette interprétation.

• Une interprétation qui satisfait un ensemble de FBFs est appelée un **modèle** de cet ensemble.

• une FBF est **valide** si elle est *vraie* pour *toutes* les interprétations des atomes qui la constituent

$P \rightarrow P$,
$P \vee \mathbf{T}$,
$\neg(P \wedge \neg P)$

Une formule est inconsistante ssi elle est fausse pour toutes les Les interpretations des atomes qui la constituent (exple: $P \wedge \neg P$)

Sémantique du Calcul Propositionnel …

- **FBFs équivalentes :** donnent la même valeur de vérité dans tous les cas possibles.

  Exemple :
  P $\rightarrow$ **Q** est équivalent à $\neg$ **P** $\vee$ **Q**

P	**Q**	$\neg$ **P**	$\neg$ **P** $\vee$ **Q**	**P** $\rightarrow$ **Q**
V	V	F	V	V
V	F	F	F	F
F	V	V	V	V
F	F	V	V	V

- **Équivalences possibles :**

$\neg(\neg P) \equiv P$

$(P \vee Q) \equiv (\neg P \rightarrow Q)$

$(P \rightarrow Q) \equiv (\neg Q \rightarrow \neg P)$: Contra-positive

$\neg(P \vee Q) \equiv (\neg P \wedge \neg Q)$ et $\neg(P \wedge Q) \equiv (\neg P \vee \neg Q)$: Règle de Morgan

$(P \vee Q) \equiv (Q \vee P)$ et $(p \wedge Q) \equiv (Q \wedge P)$: Commutativité

$((P \wedge Q) \wedge R) \equiv (P \wedge (Q \wedge R))$
$((P \vee Q) \vee R) \equiv (P \vee (Q \vee R))$ } : Associativité

$(P \vee (Q \wedge R)) \equiv (P \vee Q) \wedge (P \vee R)$
$(P \wedge (Q \vee R)) \equiv (P \wedge Q) \vee (P \wedge R)$ } : Distributivité

Système complet de Connecteurs

- Un ensemble S de symbole de connecteurs est dit Système complet de connecteurs si et seulement si toute formule *F* est equivalente à une formule G écrite en utilisant seulement les connecteurs de l'ensemble S.
- Les systèmes $\{\neg, \vee\}$,$\{\neg, \wedge\}$, $\{\neg, \rightarrow\}$, $\{\uparrow\}$,$\{\downarrow\}$

Sont des systèmes complet de connecteurs

$\uparrow$ se lit nand et est défini par $p \uparrow q = \neg (p \wedge q)$

$\downarrow$ se lit nor et est défini par $p \downarrow q = \neg (p \vee q)$

Plan

- Introduction
- Logique des propositions
  - **Syntaxe**
  - **Sémantique**
  - **Aspects Déductifs**
- Logique du premier ordre
- Logiques non classiques

Aspects Déductifs ...

Conséquence logique (entailment)

• si une FBF A a une valeur vraie pour toutes les interprétations pour lesquelles les FBFs de l'ensemble W sont vraies, alors nous disons que A est une **conséquence logique** de W.

$$W \models A$$

- $\{P, P \rightarrow Q\} \models Q$
- $\mathbf{F} \models A$ (A n'importe quelle FBF)
- $P \wedge Q \models P$

Si certaines propositions sont vraies alors d'autres sont nécessairement vraies.

Système formel

Un sytéme formel S est la donnée de :

- Un ensemble dénombrable de symboles
- Un ensemble de formules
- Un ensemble d'axiomes
- Un ensemble fini de régles d'inférences

Aspects Déductifs

• Une **règle d'inférence** est une manière de générer des FBFs à partir d'autres FBFs

  • La FBF B peut être inférer à partir des FBFs A et A $\rightarrow$ B *(modus ponens)*

  • La FBF A $\wedge$B peut être inférer de A et B (introduction de $\wedge$**)**

  • La FBF B $\wedge$ A peut être inférer de A $\wedge$B (commutativité)

  • La FBF A peut être inférer de $\neg(\neg$ A) (élimination de $\neg$)

Définition de preuve

• La séquence de FBFs $\{A_1, A_2, \ldots, A_n\}$ est appelée une **preuve** de A_n à partir d'un ensemble de FBFs W ssi chaque A_i de la séquence appartient à W ou bien il peut être inférer d'une FBF antérieure en utilisant les règles d'inférence.

• S'il existe une preuve de A_n à partir de W, nous disons que A_n est un **théorème** de l'ensemble W.

$$W \vdash_R A_n$$

On peut inférer A_n à partir de W en utilisant les règles d'inférence R

Correction (soundness)

- Pour tout ensemble de FBFs W, et pour toute FBF A

$$W \vdash_R A \rightarrow W \models A$$

Complétude

- Pour tout ensemble de FBFs W, et pour toute FBF A

$$W \models A \rightarrow W \vdash_R A$$

Quelques Résultats

- Si W_1 inclue W_2 et $W_1 \vdash A$ Alors $W_2 \vdash A$
- Si $W_1 \vdash A$ et si pour tout énoncé B de W_1 on a $W_2 \vdash B$ alors $W_2 \vdash A$

Système formel pour la LP

- On se restreint au système complet de connecteurs $\{\neg, \rightarrow\}$.
- On se donne trois schémas d'axiome définissant un ensemble infini A d'axiomes

SA1: $(A \rightarrow (B \rightarrow A))$

SA2: $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$

SA2: $(\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A)$

On utilise une seule règle de déduction:

Le modus ponens : $A, A \rightarrow B \vdash B$

Principaux Résultats

- $\vdash A \rightarrow A$

1. SA2: $(A \rightarrow ((A \rightarrow A) \rightarrow A)) \rightarrow ((A \rightarrow (A \rightarrow A)) \rightarrow (A \rightarrow A))$
2. SA1: $(A \rightarrow ((A \rightarrow A) \rightarrow A))$

3. MP 1 et 2 : $(A \rightarrow (A \rightarrow A)) \rightarrow (A \rightarrow A)$
4. SA1: $A \rightarrow (A \rightarrow A)$
5. MP 3 et 4 $A \rightarrow A$

- $\models A \rightarrow B$ si et seulement si $A \models B$

Résolution

• La résolution est une règle d'inférence qui combine plusieurs autres règles d'inférence (incluant modus ponens)

• La règle de résolution pour le calcul propositionnel s'écrit:

De $A \vee B$ et $\neg A \vee C$ nous pouvons inférer $B \vee C$,

Où A, B et C sont FBFs

La résolution est une règle d'inférence correcte

Résolution

Définitions

- Un **literal** est un atome ou la négation d'un atome P, $\neg P$
- Une **clause** est une disjonction de littéraux, $P \vee \neg Q \vee \ldots \vee R$

Généralisation de la résolution

De $P_1 \vee P_2 \vee \ldots \vee P_n$ et $\neg P_1 \vee Q_1 \vee \ldots \vee Q_k$
nous pouvons inférer $P_2 \vee \ldots \vee P_n \vee Q_1 \vee \ldots \vee Q_k$

Résolution

- **Exemples**

1. Résolution de $R \vee P$ et $\neg P \vee Q$ donne $R \vee Q$

- $R \vee P$ est équivalent a $\neg R \rightarrow P$
- $\neg P \vee Q$ est équivalent a $P \rightarrow Q$
- $R \vee Q$ est équivalent a $\neg R \rightarrow Q$

Ce qui est équivalent à la règle d'inférence par chaînage

2. Résolution de R et $\neg R \vee P$ donne P

Ce qui est équivalent à la règle d'inférence du modus ponens

Formes Normales

- Une formule F est sous forme normale conjonctive (**FNC**) ssi elle est composée de conjonctions de disjonctions de litéraux
- Une formule F est sous forme normale disjonctive (FND) ssi elle est composée de disjonctions de conjonctions de litéraux
- Exemple: $(p \vee q) \wedge (p \vee r)$ est une forme normale conjonctive

Intérêt des formes normales

- Toute formule du calcul propositionnel est équivalente à une formule sous forme normale conjonctive
- Toute formule du calcul propositionnel est équivalente à une formule sous forme normale disjonctive

Formes Normales

- Construction d'une forme normale en utilisant la table de vérité

p	**q**	**r**	**F**
V	V	V	**V**
V	V	F	**F**
V	F	V	**V**
V	F	F	**V**
F	V	V	**F**
F	V	F	**F**
F	F	V	**F**
F	F	F	**T**

Conjonction des lignes qui donnent F.
Pour chaque ligne: Disjonction des litéraux avec la négation des litéraux qui sont V.

L2: $\neg p \vee \neg q \vee r$
L5: $p \vee \neg q \vee \neg r$
L6: $p \vee \neg q \vee r$
L7: $p \vee q \vee \neg r$

FNC: $(p \vee \neg q \vee r) \wedge (p \vee \neg q \vee \neg r) \wedge (p \vee \neg q \vee r) \wedge (p \vee q \vee \neg r)$

Formes Normales

- Construction d'une forme normale en utilisant la table de vérité

p	**q**	**r**	**F**
V	V	V	**V**
V	V	F	**F**
V	F	V	**V**
V	F	F	**V**
F	V	V	**F**
F	V	F	**F**
F	F	V	**F**
F	F	F	**T**

Disjonction des lignes qui donnent V.
Pour chaque ligne: Conjonction des litéraux avec la négation des litéraux qui sont F.

L1: $p \wedge q \wedge r$
L3: $p \wedge \neg q \wedge r$
L4: $p \wedge \neg q \wedge \neg r$
L8: $\neg p \wedge \neg q \wedge \neg r$

FND: $(p \wedge q \wedge r) \vee (p \wedge \neg q \wedge r) \vee (p \wedge \neg q \wedge \neg r) \vee (\neg p \wedge \neg q \wedge \neg r)$

Formes Normales

- Simplification
- $A \equiv (A \vee B) \wedge (A \vee \neg B)$
- $A \equiv (A \wedge B) \vee (A \wedge \neg B)$

Résolution

Conversion d'une FBF à une conjonction de clauses (FNC)

$$\neg (P \rightarrow Q) \vee (R \rightarrow P)$$

1. Éliminer le signe d'implication

$$\neg (\neg P \vee Q) \vee (\neg R \vee P)$$

2. Réduire la portée de la négation en utilisant la loi de DeMorgan et éliminer les doubles négations

$$(P \wedge \neg Q) \vee (\neg R \vee P)$$

3. Convertir à un ensemble de conjonction en utilisant les lois d'associativité et de distributivité

$$(P \vee \neg R) \wedge (\neg Q \vee \neg R \vee P)$$

Ce qui donne l'ensemble des clauses (reliées par conjonction)

$\{(P \vee \neg R), (\neg Q \vee \neg R \vee P)\}$

Résolution

La résolution n'est pas une règle d'inférence complète

On ne peut pas trouvé une inférence pour $P \wedge R \models P \vee R$
en utilisant la résolution pour l'ensemble des clauses {P,R}

→ **Réfutation**

Utiliser la résolution pour montrer que la négation de $P \vee R$
est inconsistante avec l'ensemble des clauses {P,R}

$\neg(P \vee R)$ donne $\neg P \wedge \neg R$ dans l'ensemble des clauses
nous auront {P,R, $\neg P$, $\neg R$ }

La résolution utilisant la réfutation est une procédure d'inférence complète (pour toute formule valide on peut trouver une inférence)

Résolution

1. Convertir les FBFs en une forme clausale (conjonction de clauses)

2 . Convertir la négation de la FBF, A, à prouver en une forme clausale

3. Combiner les clauses résultant des 2 étapes précédentes dans un ensemble W

4. Appliquer itérativement la résolution pour les clauses dans W Jusqu'à il n'y pas de résultat à ajouter ou la clause vide est produite

Résolution

Exemple

Calcul de Prédicats (ou Logique de 1er Ordre)

Calcul propositionnel : chaque atome (P, Q, R, …) dénote une proposition d'une certaine complexité.

Modéliser:
Toutes les chandelles sont faites pour éclairer
Quelques chandelles éclairent très mal
Quelques objets qui sont fait pour éclairer le font très mal

Calcul de Prédicats (ou Logique de 1er Ordre) …

- Notion de relation (unaire, binaire, . . . , n-aire)
- Notion de variable (« place » réservée pour un énoncé à venir)
- Notion de fonction
- Notion de quantification

- **Calcul de prédicats :** permet de généraliser les assertions moyennant les variables.

  Exemple :

  Calcul propositionnel : p = « il a plu mardi »

  Calcul de prédicats : temps(mardi, pluie)

  prédicat décrivant la relation entre une date et le temps.

  Généralisation ⟶ pour toutes les valeurs de X, où X est un jour, le prédicat weather(X, rain) est vrai.

- Le calcul de prédicats permet de s'intéresser à des énoncés relatifs à un ensemble d'objets ou certains éléments de cet ensemble.

Syntaxe du Calcul de Prédicats …

- **Alphabets** utilisés pour construire les symboles :
  - a, b, …, z, A, B, …, Z (lettres majuscules et minuscules de l'alphabet)
  - 0, 1, …, 9 (les chiffres)
  - _ (caractère souligné)

Symboles légaux	**Symboles non légaux**
George	3jack
fire3	no blanks allowed
tom_and_jerry	ab%cd
bill	***71
XXXXX	duck??
friends_of	

l(g, k) —Améliorer la lisibilité→ likes(george, kate)

Syntaxe du Calcul de Prédicats (suite) …

- **Constantes :** symboles commençant par une lettre minuscule.
  george, tree, geORGE, …
- **Variables :** symboles commençant par une lettre majuscule.
  George, Bill, Kate, …
- **Fonctions :** symboles commençant par une lettre minuscule.
  Chaque fonction est d'arité positive ou nulle (nombre de paramètres).

  plus(5, 2) : arité 2
  father(george) : arité 1
  price(bananas)
  f(X, Y)
- **Connecteurs :** ceux du calcul propositionnel
- **Quantificateurs :** ∀ (universel) et ∃ (existentiel)

  ∃ Y friends(Y, peter)
  ∀ X likes(X, ice_cream)

Syntaxe du Calcul de Prédicats (Suite) …

- **Terme :** défini récursivement par :
  - Une variable est un terme.
  - Un symbole de constante est un terme.
  - **Si** f est un symbole de fonction d'arité n et si t_1, t_2, …, t_n sont des termes, **alors** f (t_1, t_2, …, t_n) est un terme.

Exemples : Cat X mother(jane)
times(2, 3) Blue Kate

- **Prédicats :** symboles commençant par une lettre minuscule.
  Chaque prédicat décrit une relation entre zéro ou plusieurs objets.

Exemples : likes Near on
equals part_of

Description d'un ensemble de relation de parenté :

mother(eve, lebel) mother(eve, cain)
father(adam, abel) father(adam, cain)
$\forall X \forall Y$ father(X, Y) $\vee$ mother(X, Y) $\rightarrow$ parent (X, Y)
$\forall X \forall Y \forall Z$ parent(X, Y) $\wedge$ parent(X, Z) $\rightarrow$ sibling(Y, Z)

Syntaxe du Calcul de Prédicats (suite) …

- **Atome : Si** p est un symbole de prédicat d'arité n et si t_1, t_2, …, t_n sont des termes, **alors** p(t_1,t_2, …,t_n) est un atome (ou formule atomique).

  Exemples :

  helps(bill, george)
  friends(father_of(david), father_of(andrew))
  likes(george, kate)
  likes(X, george)
  likes(X, X)
  likes(george, sarah, tuesday)

- Un symbole de prédicat peut être utilisé avec différents nombres de paramètres (représente dans ce cas des relations différentes).

Syntaxe du Calcul de Prédicats (suite) …

☯ **Formule bien formée (FBF) :** définie récursivement par :

- Tout atome est une formule.
- **Si** A et B sont deux formules, **alors** $(A \rightarrow B)$, $(A \equiv B)$, $(A \wedge B)$ et $(A \vee B)$ sont des formules.
- **Si** A est une formule, **alors** $\neg A$ est une formule.
- **Si** A est une formule, et si X est une variable quelconque, **alors** $(\forall X\ A)$ est une formule.
  (On dit que A est la portée du quantificateur $\forall X$)
- **Si** A est une formule, et si X est une variable quelconque, **alors** $(\exists X\ A)$ est une formule.
  (On dit que A est la portée du quantificateur $\exists X$)

Syntaxe du Calcul de Prédicats (suite) …

- Différentes relations entre la négation et le quantificateurs ∃ (existentiel) et ∀ (universel) :

  $\neg \exists X\ p(X) \equiv \forall X \neg p(X)$

  $\neg \forall X\ p(X) \equiv \exists X \neg p(X)$

  $\exists X\ p(X) \equiv \exists Y\ p(Y)$: variable muette

  $\forall X\ q(X) \equiv \forall Y\ q(Y)$

  $\forall X\ (p(X) \wedge q(X)) \equiv \forall X\ p(X) \wedge \forall Y\ q(Y)$

  $\exists X\ (p(X) \vee q(X)) \equiv \exists X\ p(X) \vee \exists Y\ q(Y)$

- La logique de prédicat de 1er ordre permet aux variables quantifiables de référer aux objets du domaine, mais pas aux prédicats et fonctions.

  $\forall$ (likes) likes(george, kate) — expression mal formée selon la logique de prédicat de 1er ordre

- D'autres logiques de prédicats d'ordre supérieur permettent de telles expressions.

Syntaxe du Calcul de Prédicats (suite) …

- **Exemples de FBFs :**

  times, plus : Fonctions d'arité 2

  equal, foo : prédicats d'arité 2 et 3 respectivement

  plus(two, three) : Fonction mais pas un atome

  equal(plus(two, three), five), equal(plus(2, 3), seven) : Deux atomes

  $\exists$ X foo(X, two, plus(two, three)) $\wedge$ (equal(plus(two, three), five)

  : Formule (conjonction de formules)

  (foo(two, two, plus(two, three))) $\rightarrow$ (equal(plus(three, two), five) $\equiv$ true)

  : Formule (tous les composants sont des formules connectées par des opérateurs logiques)

Calcul de Prédicats : Exemples …

- If it doesn't rain on Monday, Tom will go to the mountains.

  $\neg$ **weather(rain, monday) $\rightarrow$ go(tom, mountains)**
- Emma is a Doberman pinscher and a good dog.

  gooddog(emma) $\wedge$ isa(emma, doberman)
- All basketball players are tall.

  $\forall$ **X (basketball_player(X) $\rightarrow$ tall(X))**
- Some people like anchovies.

  (person(X) $\wedge$ likes(X, anchovies)
- If wishes were horses, beggars would ride.

  equal(wishes, horses) $\rightarrow$ ride(beggars)
- Nobody likes taxes.

  $\neg\exists$ **X likes(X, taxes)**

Exercice

Modéliser les expressions suivantes :

1. tous les lions sont féroces
2. quelques lions ne boivent pas de café
3. aucun singe n'est soldat
4. tous les singes sont malicieux

Quelques expressions courantes

- tous les A sont B $\forall x, A(x) \rightarrow B(x)$
- seuls les A sont B $\forall x, B(x) \rightarrow A(x)$
- aucun A n'est B $\forall x\ A(x) \rightarrow \neg(B(x))$
- quelques A sont B $\exists x, A(x) \wedge B(x)$

Règles d'Inférences : Définitions …

- **Règle d'inférence :** moyen permettant de produire de nouvelles expressions du calcul de prédicats à partir d'autres expressions.

- Règles d'inférence utiles :
  - **Modus ponens** : si les deux expressions P et P $\rightarrow$ Q sont vraies, alors Q est vraie.

P	**P $\rightarrow$ Q**	**Q**
V	**V**	**V**
V	F	F
F	V	V
F	V	F

Règles d'Inférences Utiles (suite) …

- **Modus ponens** (suite) :

Observations : 1) « If it is raining then the ground will be wet »
2) « it is raining »

P = « It is raining »
Q = « The ground is wet »

Observation 1) ⟶ P → Q est vraie

Ensemble d'axiomes étant donné qu'il pleut maintenant :

P
P → Q

Modus ponens ⟶ On peut ajouter le fait que la terre est mouillée (Q) à l'ensemble des expressions ayant la valeur vraie.

Règles d'Inférences Utiles (suite) …

- **Modus tollens** : si P $\rightarrow$ Q est vraie et Q est fausse, alors $\neg$ P est vraie (ou P est fausse).

P	P $\rightarrow$ Q	Q
V	V	V
V	F	F
F	V	V
F	**V**	**F**

- **Élimination « et »** : si P $\wedge$ Q est vraie, alors P est vraie et Q est vraie.

- **Introduction « et »** : si P et Q sont vraies, alors P $\wedge$ Q est vraie.

Règles d'Inférences Utiles (suite) …

- **Instantiation universelle** : si a est une constante appartenant au domaine de X et que ∀ X p(X) est vraie, alors p(a) est vraie.

« All man are mortal » ⟶ ∀ X (man(X) → mortal(X))

« Socrates is a man » ⟶ man(socrates)

man(socrates) —**Instantiation universelle**→ mortal(socrates)

man(socrates) → mortal(socrates)
man(socrates)
} —**Modus ponens**→ mortal(socrates)

Cet axiome peut être ajouté à l'ensemble des expressions qui sont des **conséquences logiques** des assertions originales.

Caractérisation des variables

On appelle **occurrence de x** dans F chaque endroit où la variable x apparaît dans la formule F non immédiatement précédée d'un symbole de quantificateur.

Occurrence libre

1. si F est un atome, toutes les occur. de x dans F sont libres
2. si F = ¬(G), les occurrences libres de F sont celles de G
3. si F = (G)*(H), où * est un symbole de connecteur binaire, les occurrences libres de x dans F sont la réunion de celles de G et de celles de H.
4. si F = ∀ y(G) ou F = ∃ y (G), avec y variable distincte de x, les occurrences libres de x dans F sont celles de G
5. si F = ∀ x(G) ou F = ∃ x(G), aucune occurrence de x dans F n'est libre.

Caractérisation des variables

- Une variable est libre (ou parlante) si elle a au moins une occurrence libre.
- NB : une variable n'ayant aucune occurrence libre est dite liée (ou muette).

Définition (Formule close)

- Une formule n'ayant pas de variable libre est dite close.

Exercice

- Pour chaque variable apparaissant dans ces formules,signaler chacune des occurrences.
- S'agit-il de variables liées ou libres
- S'agit-il de formules closes ?

1. $\exists\, y(R(x, f(y), z) \rightarrow (v(b) \vee s(a, g(b))))$
2. $(\forall x\, p(x)) \vee_ (\exists\, y\, f(x) = y)$
3. $\forall x\, \forall y\, (x = y \rightarrow f(x) = f(y))$

Sémantique du Calcul de Prédicats …

- **Objectif :** Construire un modèle permettant de dégager une interprétation sémantique des formules.
- **Sémantique du calcul des prédicats :** fournit une base pour déterminer la valeur de vérité des expressions bien formées.

  Exemple :

  friends(george, susie)
  friends(george, kate)

  } Valeur de vérité de « *george est un ami de susie* » est vrai (V)

- Calcul propositionnel : utilisation des tables de vérité.
- Calcul de prédicats : différents domaines de variables

  → **problème :** Impossible d'utiliser la méthode de tables de vérité directement dérivée du calcul propositionnel.

  Solution : Utiliser une fonction d'interprétation.

Sémantique du Calcul de Prédicats (suite) …

Formule atomique $p(X_1, \ldots, X_n)$ —Donner un sens au symbole p→ Valeur de vérité selon les valeurs possibles des $X_1, \ldots, X_n$

☯ **Exemple :**

prédicat p d'arité un : est_bleu
domaine : {martien, Homme, schtroumfs}
Interprétation possible, notée I(p)(X), de p est :

X	I(p)(X)
martien	F
Homme	F
schtroumfs	V

Sémantique du Calcul de Prédicats (suite) …

☯ Interprétation d'une formule F est caractérisée par la donnée de :

- Un ensemble D ($\neq \emptyset$), appelé *domaine de l'interprétation*.
- Une fonction d'interprétation I qui associe à :

Chaque symbole de F ⟶ Un élément de D

Constante ⟶ un élément de D

Variable ⟶ un sous-ensemble de D

Chaque symbole de fonction f d'arité n ⟶ Graphe d'une fonction de $D^n \rightarrow D$ définissant f

Chaque symbole de prédicat p d'arité n ⟶ Graphe d'une fonction de $D^n \rightarrow \{V, F\}$ définissant p

Exemple

- $HG = \forall x \forall y \forall z (P(x, y) \wedge P(y, z)) \rightarrow G(x, z)$
- $HP = \forall x \exists y P(y, x)$
- $C = \forall x \exists y G(y, x)$
- $D = \forall x \forall z P(z, f(x)) \rightarrow G(z, x)$
- $F = HG \wedge HP \rightarrow C$

Exemple

D comme est l'ensemble des êtres humains. La relation P(x, y) signifie que x est le père de y. La relation G(x, y) signifie que x est un grand-père de y. La fonction f associe un individu à sa mère.

Exercice

Comment lire les formules ?

Interprétation de Formules …

- Soit : E une expression I une interprétation de E sur le domaine D
- Pour une interprétation I donnée, on appelle **valuation** v des variables relatives à I, toute application de l'ensemble des variables dans D.

Interprétation de Formules …

☯ Soit : E une expression
I une interprétation de E sur le domaine D
La valeur de vérité de E est déterminée par :

1) Symbole de constante ⟶ Sa valeur selon I

2) Variables (libres) ⟶ Valeurs de D affectée à ces variables par I (valuation v)

3) Symbole de fonction $f(t_1, \ldots, t_n)$ ⟶ Terme $f'(\underbrace{t'_1, \ldots, t'_n}_{\text{Interprétations des } t_1, \ldots, t_n})$

Interprétation de f

4) Symbole de vérité { « vrai » | « faux » } ⟶ V | F

5) Atome $p(t_1, \ldots, t_n)$ ⟶ Fonction $p'(\underbrace{t'_1, \ldots, t'_n}_{\text{Interprétations des } t_1, \ldots, t_n})$

Interprétation de p

Interprétation de Formules (suite) …

6-10) E est de la forme ($\neg$ A), ($A \rightarrow B$), ($A \equiv B$), ($A \wedge B$) ou ($A \vee B$) ⟶ I(E) est définie par les mêmes lois fonctionnelles que celles du calcul propositionnel

11) E est de la forme $\forall X\, g(X, Y_1, \ldots, Y_n)$, pour tout n-uplet $(a_1, \ldots, a_n) \in D^n$ ⟶ $I(E)(a_1, \ldots, a_n) = T$ si pour tout valeur $a \in D$, $I(g)(a, a_1, \ldots, a_n) = T$

$I(E)(a_1, \ldots, a_n) = F$ sinon.

12) E est de la forme $\exists X\, g(X, Y_1, \ldots, Y_n)$, pour tout n-uplet $(a_1, \ldots, a_n) \in D^n$ ⟶ $I(E)(a_1, \ldots, a_n) = T$ S'il existe une valeur $a \in D$, $I(g)(a, a_1, \ldots, a_n) = T$

$I(E)(a_1, \ldots, a_n) = F$ sinon.

- **Formule valide :** une formule F est dite universellement valide si et seulement si, pour toute interprétation I, on a I(F) = T.

  $p(X) \vee \neg p(X)$: formule valide

- **Formule satisfiable et formule inconsistante :**
  une formule A sera dite satisfiable, ou sémantiquement consistante, s'il existe une interprétation I telle que I(A) = T.

  Domaine d'interprétation = {1, 2, …}
  A = inferieur_a(X, 0) formule inconsistante (non satisfiable)
  B = superieur_a(X, 0) formule satisfiable
  $C = \exists X (p(X) \wedge \neg p(X))$ formule inconsistante

- **Conséquence logique :** formule B, famille de n formules A_i
  B est conséquence logique des A_i si pour toute interprétation I telle que $\forall A_i$, $I(A_i) = T$, on a aussi I(B) = T.

  Domaine d'interprétation = {2, 4, 6, …}
  A_1 = pair(X)
  A_2 = superieur_a(X, 0)
  B = ¬ pair(plus(X, 1)) ∧ ¬ superieur_a(moins(X, plus(X, X)),0)

Unification …

- Pour appliquer les règles d'inférence telle que modus ponens, un système d'inférence doit être capable de déterminer quand deux expressions sont identiques ou se correspondent.
- **Cas du calcul propositionnel :** deux expressions se correspondent SSI elles sont syntaxiquement identiques.
- **Cas du calcul des prédicats :** le processus de vérifier si deux expressions se correspondent est plus complexe (du fait de l'existence des variables dans les expressions).
- **Unification :** algorithme permettant de déterminer quelles substitutions sont nécessaires pour faire correspondre deux expressions.

Exemple 1 :

$\forall X\ man(X) \rightarrow mortal(X)$
$man(socrates)$

Substituer X par socrates
et utiliser modus ponens
⟶ mortal(socrates)

Unification (suite)…

Exemple 2 : variables muettes (non quantifiées)

p(X) et p(Y) sont équivalentes ⟶ On peut substituer Y par X pour faire correspondre les deux expressions.

Unification **+** Règles d'inférences ⟶ Faire des inférences sur un ensemble d'assertions logiques.

La base de données doit être exprimée sous une forme appropriée.

- **Cas du quantificateur ∃ :** une variable quantifiée existentiellement est éliminée des expressions de la base de données en la remplaçant par une constante qui donne la valeur vraie à ces expressions.

∃ X parent(X, tom) ⟶ parent(bob, tom) ou parent(mary, tom)
Où bob et mary sont des parents de tom selon l'interprétation considérée.

Unification (suite)…

- **Problème :** Cas où la valeur des substitutions dépend de la valeur d'autres variables de l'expression :

  $\forall X \exists Y$ mother(X, Y) ⟶ La valeur de la variable quantifiée existentiellement Y dépend de la valeur de X

- **Solution :** Skolémisation
  Construire une formule dont la consistance est équivalente à la consistance de la formule initiale.

- **Algorithme de Skolémisation :** permet de construire la forme de Skolem S_F d'une formule F :

  1) Transformer la formule F en une forme prénexe.

  2) Remplacer toutes les variables quantifiées existentiellement par un symbole de fonction dont les arguments sont les variables qui sont quantifiées universellement et qui précèdent notre variable.

  3) Supprimer les quantificateurs existentiels devenus inutiles.

- Exemple.

Unification (suite)…

- Suppression des quantificateurs existentielles.

  → L'unification peut être utilisée pour trouver une correspondance entre deux formules afin d'appliquer les règles d'inférence telle que modus ponens.

- Exemple d'unification : soit l'expression foo(X, a, goo(Y))

  1) foo(fred, a, goo(Z)) ⟶ {fred/X, Z/Y}

  2) foo(W, a, goo(jack)) ⟶ {W/X, jack/Y}

  3) foo(Z, a, goo(moo(Z))) ⟶ {Z/X, moo(Z)/Y}

  Exemples d'unification permettant de rendre l'expression initiale Identique à l'une des trois expressions

- Une variable ne peut être unifiée avec un terme contenant cette variable.

  ~~p(X)/X~~

  Faux, parce que cette substitution crée une expression infinie : p(p(p(… X) …)

Substitution …

- **Substitution par renommage :** si X_1/X_2 est plus tard X_1 est remplacée par une constante, alors il faut que la variable X_2 reflète cette substitution.

  ① $p(Y, Z) \rightarrow q(Y, Z)$ ② $q(W, b) \rightarrow r(W, b)$

  Substitution 1 = {a/Y, X/Z} dans clause ① ⟶ q(a, X)

  Substitution 2 = {a/W, b/X} dans clause ② ⟶ r(a, b)

- **Composition de substitutions :** soit S et S' deux substitutions définies par :

  $S = \{t_1/X_1, \ldots, t_n/X_n\}$

  $S' = \{u_1/Y_1, \ldots, u_n/Y_n\}$

Substitution (suite)…

- **Composition de substitutions :** la composition de S et S' est la substitution SS' définie à partir de l'ensemble

$$\{\underbrace{S'(t_1)/X_1, ..., S'(t_n)/X_n}_{\text{Appliquer la substitution S' aux éléments de S}}, \underbrace{u_1/Y_1, ..., u_n/Y_n}_{\text{S'}}\}$$

  dont on a supprimé tous les $S'(t_i)/X_i$ pour lesquels $X_i = S'(t_i)$ et tous les u_i/Y_i tel que $Y_i \in \{X_1, \ldots, X_n\}$

- Exemples.

- **Unificateur :** soient deux expressions E_1 et E_2.
  On appelle unificateur de E_1 et E_2 toute substitution S telle que $E_1S = E_2S$.

  Exemple : E_1 = p(X) E_2 = p(Y)
  Unificateur 1 : {fred/X, fred/Y}
  Unificateur 2 : {Z/X, Z/Y}

Unificateur le Plus Général …

- **Unificateur le Plus Général** d'un ensemble d'expressions E, un unificateur g de E (s'il existe) tel que pour tout unificateur S de E il existe une substitution S' tel que E S = E g S'.

  Exemple : E_1 = Y E_2 = f(X)
  Unificateur le plus général : {f(X)/Y}

- **Algorithme d'unification :**
  Simplification de la syntaxe :

Syntaxe du calcul de prédicats	**Syntaxe sous forme de listes**
p(a, b)	(p a b)
p(f(a), g(X, Y))	(p (f a) (g X Y))
equal(eve, mother(cain))	(equal eve (mother cain))

Algorithme d'Unification …

function unify(E_1, E_2) permet d'unifier deux expressions
begin
  case
    both E_1 and E_2 are constants or the empty list:
      if (E_1 = E_2)
      then return {}
      else return FAIL;
    E_1 is a variable:
      if (E_1 occurs in E_2)
      then return FAIL
      else return {E_2 / E_1};
    E_2 is a variable:
      if (E_2 occurs in E_1)
      then return FAIL
      else return {E_1 / E_2};
    either E_1 or E_2 is the empty list: return FAIL

Algorithme d'Unification (suite) …

otherwise:

begin

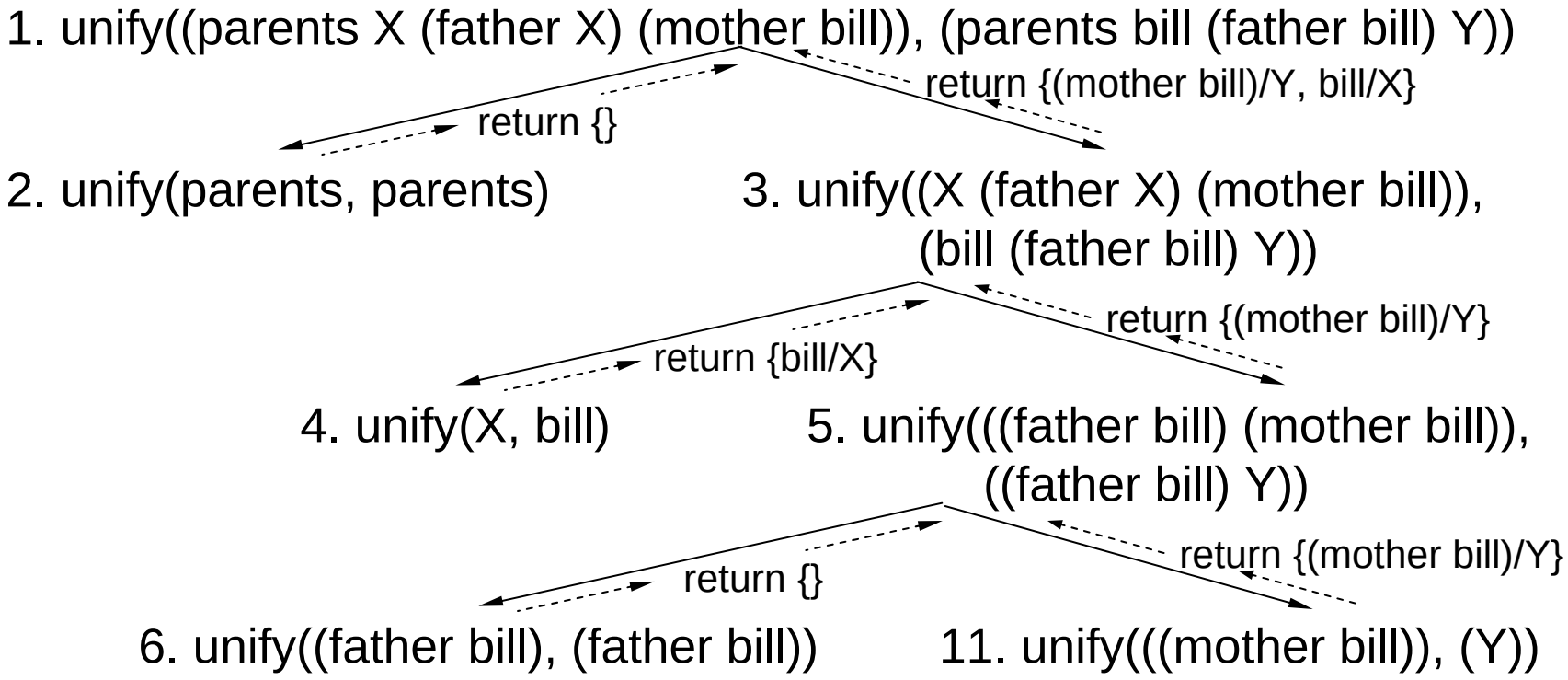
HE_1 := first element of E_1;
HE_2 := first element of E_2;
$SUBS_1$:= unify(HE_1, HE_2);
if ($SUBS_1$ = FAIL) **then** return FAIL;
TE_1 := apply ($SUBS_1$, rest of E_1);
TE_2 := apply ($SUBS_1$, rest of E_2);
$SUBS_2$:= unify(TE_1, TE_2);
if ($SUBS_2$ = FAIL)
then return FAIL;
else return composition ($SUBS_1$, $SUBS_2$);

end

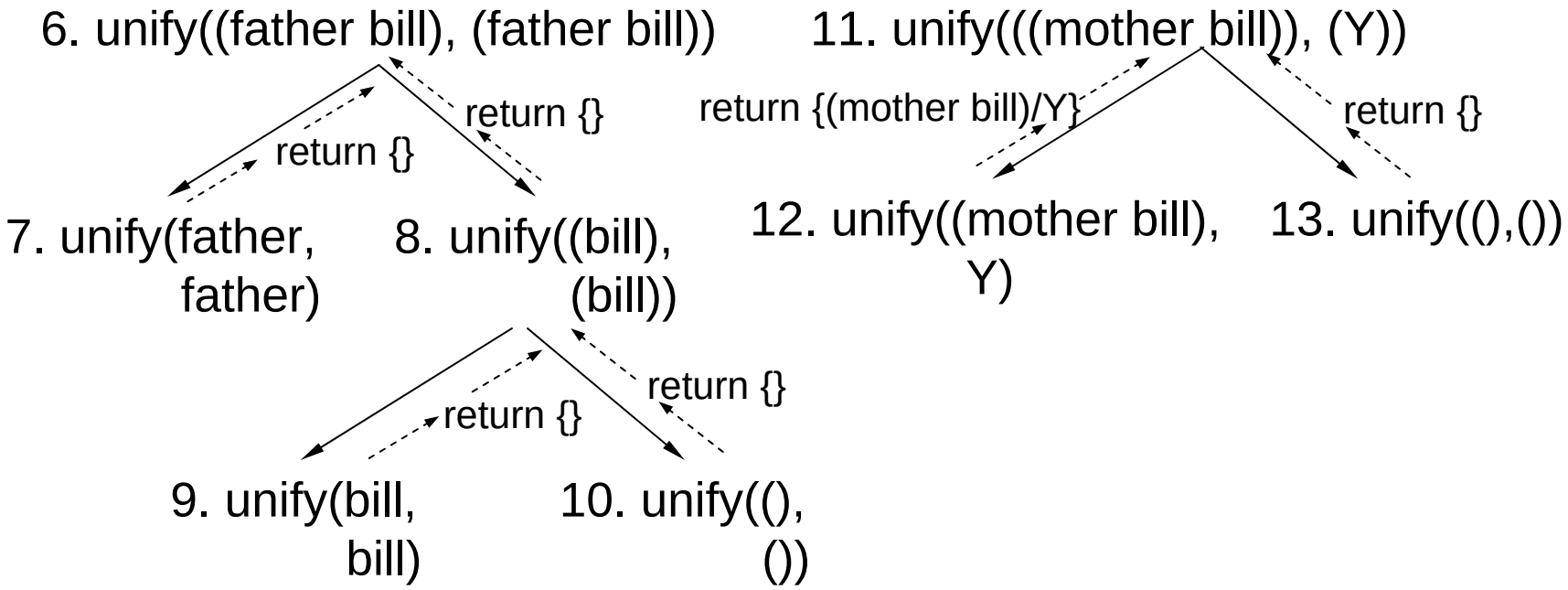
end

end.

Exemple d'Unification …



Exemple d'Unification (suite) …



☯ Exemples.