2 Syntaxe de la logique des prédicats

Eléments de vocabulaire

— *syntaxe* ce qu'on a le droit d'écrire
— *sémantique* ce que cela signifie, comment on *interprète* une phrase, comment on définit $\models$
— *calcul* des mécanismes qui permettent de garantir que sont cohérents des raisonnements comme par exemple $H,\ H \rightarrow C\ \models\ C$

intuition sur le premier ordre

— **autorisé**
la relation $\mathcal{R}$ est transitive se modélise par
$\forall x\ \forall y\ \forall z\ \mathcal{R}(x,y) \wedge \mathcal{R}(y,z) \rightarrow \mathcal{R}(x,z)$
— **interdit**
$\forall \mathcal{R} :\ Transitive(\mathcal{R}) \leftrightarrow \{\forall x \forall y \forall z \mathcal{R}(x,y) \wedge \mathcal{R}(y,z) \rightarrow \mathcal{R}(x,z)\}$
— **autorisé**
$\forall \mathcal{R}\, \forall \mathcal{R}'\ Bijection(\mathcal{R}) \wedge Transitive(\mathcal{R}) \wedge Inverse(\mathcal{R},\mathcal{R}') \rightarrow Transitive(\mathcal{R}')$
— **interdit dans cette première partie**
$\forall \mathcal{R}\ Bijection(\mathcal{R}) \wedge\ Transitive(\mathcal{R}) \rightarrow Transitive(Inverse(\mathcal{R}))$

2.1 Quelques remarques préalables

vocabulaire de base à connaître

Pour construire des phrases logiques (formules logiques, **fbf**) on disposera

— des termes : ils servent à désigner des objets du domaine modélisé ; ils sont de deux sortes
  — des constantes : $a,\ b,\ c \ldots$
  — des variables $x,\ y,\ z \ldots$
— des symboles de prédicat $P,\ Q,\ R \ldots$ comme notation abstraite, $Vert,\ PereDe,\ Donne \ldots$ dans les modélisations
un symbole de prédicat a une *arité* (resp. 1, 2 et 3 pour $Vert,\ PereDe,\ Donne$)
— des connecteurs logiques $\wedge \vee \ldots$
— des quantificateurs $\forall,\ \exists$
— des parenthèses (un seul jeu suffit, plusieurs comme sucre syntaxique).

intuition sur l'aspect formel :

Les plus petites formules, les *atomes* sont formées d'un symbole de prédicat et de ses arguments qui sont des termes : $Prof(x)$, $Cours(l_{gqe}$, $DonneCours(d, x)$. Ces atomes sont complexifiés grace aux quantificateurs et combinés entre eux grace aux connecteurs logiques de façon récursive :
$\exists x\ [Prof(x) \wedge DonneCours(x, l_{gqe})]$ (on combine deux atomes puis on complexifie)
$\forall x\ \{[\exists y\ :\ P(y) \wedge Q(x, y)] \rightarrow [\exists z\ :\ S(x, z)]\}$ là c'est très complexifié

précisions sur l'aspect modélisation

pour une modélisation donnée, on aura un ensemble bien précis de constantes et de symboles de prédicat

- Adam, Eve, Homme(.), Femme(.), FilsDe(.,.)
- Tom, Brad, Acteur(.), JoueDans(.,.)

Par contre les variables, les connecteurs logiques, les quantificateurs, les parenthèses sont les mêmes pour toutes les modélisations. C'est pour cela qu'on définira un *langage* à partir de ses constantes et ses prédicats.

dans la deuxième partie

les *termes* peuvent être complexes : par exemple $f(x, g(a, y))$ où f et g sont des fonctions d'arité 2.

- $\forall x \forall y\ Pair(x) \wedge Pair(y) \rightarrow Pair(Plus(x, y))$ versus
- $\forall x \forall y\ Pair(x) \wedge Pair(y) \rightarrow \exists z\ Pair(z) \wedge EstSomme(z, x, y)$

arité 0

Un prédicat d'arité 0 correspond à un symbole propositionnel : $IlPleut$ ou $IlPleut()$
De même une constante est un symbole de fonction 0-aire : a ou $a()$

2.2 langage du premier ordre

Remarque

En terme de langage formel, nous allons définir un *alphabet* sur lequel nous définirons ensuite une grammaire pour définir un langage. Mais traditionnellement on appelle *langage du premier ordre* cet alphabet.

Définition d'un langage du premier ordre

C'est un couple $\mathcal{L} = \mathcal{C}\ \cup\ \mathcal{P}$ de symboles constitué :

- d'un ensemble $\mathcal{C}$ de constantes
- d'un ensemble $\mathcal{P}$ de symboles de prédicats avec une arité associée : $\{P_1, Q_2, R_2 \ldots\}$
- $\mathcal{C} \cap \mathcal{P} = \emptyset$

Termes

En plus des constantes, on dispose d'un ensemble infini $\mathcal{V}$: $\{x, y, z \ldots\}$ de variables disjoint de $\mathcal{C}$ et de $\mathcal{P}$: et de façon optionnelle de $\{\top, \bot\}$, deux symboles spéciaux (qui ne sont ni dans $\mathcal{V}$ ni dans $\mathcal{C}$ ni dans $\mathcal{P}$).
L'ensemble des termes d'un langage $\mathcal{L} =_{def} \mathcal{C}\ \cup\ P$ est l'ensemble
$\mathcal{T} =_{def} \mathcal{V}\ \cup\ \mathcal{C}$

Remarques

mais ça, c'est de l'interprétation, on le verra plus au cours suivant

Un terme est une expression désignant un objet du domaine modélisé

- Les constantes désignent des objets précis du domaine
- Les variables désignent des objets indéfinis du domaine
- $\top$ et $\bot$ désignent respectivement une propriété toujours vrai (ou toujours fausse) comme il peut en exister sur tout domaine.

Dans le cadre de cette introduction les termes sont élémentaires ; en logique des prédicats les termes peuvent être complexes : par exemple $f(x, g(a, y))$ où f et g sont des fonctions d'arité 2.

2.3 Formules Bien Formées

Soit $\mathcal{L} =_{def} \mathcal{C} \cup P$ un langage, $\mathcal{V}$ un ensemble infini de variables,
$K =_{def} \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ les connecteurs, $\mathcal{Q}_{tf} =_{def} \{\forall, \exists\}$ les quantificateurs, et $\mathcal{P} =_{def} \{(,)\}$ un jeu de parenthèses.

On définit par induction $FBF(\mathcal{L})$, l'ensemble des formules bien formées, construites sur $\mathcal{L}$:

- base
  1. $FBF(\mathcal{L})$ contient l'ensemble des atomes
     $\{\ p(t_1, \ldots, t_n) \mid p \in \mathcal{P}$ est un prédicat n-aire et $t_1, \ldots, t_n \in \mathcal{T}$ sont n termes $\}$
  2. $FBF(\mathcal{L})$ contient $\{\top, \bot\}$ dans la mesure où ces symboles sont admis.
  3. **le cas spécial du symbole d'égalité** $=$
     $FBF(\mathcal{L})$ contient l'ensemble des formules $\{t_1 = t_2$ avec t_1 et $t_2 \in \mathcal{C}_{ste} \cup \mathcal{V}\}$

- induction : soit A et $B \in FBF(\mathcal{L})$ et soit $x \in \mathcal{V}$:
  - $\neg A \in FBF(\mathcal{L})$
  - $(A \wedge B) \in FBF(\mathcal{L})$ (idem avec $(A \vee B)$, $(A \rightarrow B)$, $(A \leftrightarrow B)$)
  - $\forall x A \in FBF(\mathcal{L})$ (idem avec $\exists x A$)

Exemple

$\forall x\ \forall y\ \neg[x = y] \rightarrow \exists z\{<(x, z) \wedge <(z, y)\}$

redondance des connecteurs

On n'a pas besoin de tous les connecteurs et de tous les quantificateurs, on pourrait par exemple se limiter au $\neg$, au $\wedge$ et au $\exists$.

$$\forall x\ A \equiv \neg\exists x\ \neg A \text{ et } \exists x\ A \equiv \neg\forall x\ \neg A$$

le $\equiv$ n'est pas un signe syntaxique mais *sémantique* : la valeur de vérité est la même pour une interprétation donnée quelconque

attention à la *portée* des quantificateurs

On verra plus tard que $\exists x\ [P(x) \wedge Q(x)] \not\equiv [\exists x\ P(x)] \wedge Q(x)$

une dernière règle syntaxique
$(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$ c'est pour ça qu'on supprime les parenthèses : $P \wedge Q \wedge R$

2.4 arbre syntaxique d'une fbf

On devrait dire *arborescence* étiquetée sur le sommets.

exemple
$\forall x\ \forall y\ \neg[x = y] \rightarrow \exists z\{< (x, z) \wedge < (z, y)\}$
les feuilles sont les atomes.

Définition par induction de l'"arborescence syntaxique d'une fbf f

- **base :** si f est un atome, ou une formule exprimant l'égalité de deux termes ($\{t_1 = t_2$ avec t_1 et $t_2 \in \mathcal{C}_{ste} \cup \mathcal{V}\}$), ou $\top$ ou $\perp$, l'arborescence logique associée à f est réduite à un sommet étiqueté par f
- **induction** si f est une fbf et $\mathcal{A}_f$ son arborescence associée (resp. f_G et $\mathcal{A}_{f_G}$, f_D et $\mathcal{A}_{f_D}$) alors
  - $\neg f$ a pour arborescence associée une arborescence étiquetée dont la racine est étiquetée par $\neg$ et l'unique sous arborescence celle associée à f
  - Qxf (où Q est un quantificateur de $\mathcal{Q}_{tf}$ et x une variable de $\mathcal{V}$) a pour arborescence associée une arborescence étiquetée dont la racine est étiquetée par $\forall x$ et l'unique sous arborescence celle associée à f
  - $(f_G\ C_b\ f_G)$ (où C_b est un connecteur logique binaire) a pour arborescence associée une arborescence étiquetée dont la racine est étiquetée par C_b, dont la sous-arborescence gauche est $\mathcal{A}_{f_G}$ et dont la sous-arborescence droite est $\mathcal{A}_{f_D}$.

2.5 Variables libres et liées

Nous voulons parler des différences et ressemblancee entre les trois formules

$\forall x\ \exists y\ P(x, y)$
$\forall x\ P(x, y)$ qu'est ce que ce y ?
$\forall x\ (\exists y\ P(x, y)) \wedge \forall x\ (P(x, y))$ et ici ?

Portée d'un quantificateur

La portée d'un quantificateur est la sous-arborescence dont le couple $<$ce quantificateur, sa variable associée$>$ étiquette la racine.

définitions des variables libres et des variables liées

- Une occurrence d'une variable x est liée dans F si dans la branche qui va de la racine à la feuille (l'atome) où se trouve cette occurrence on trouve $\forall x$ ou $\exists x$ (autrement dit quand elle est dans la portée d'un $\forall x$ ou d'un $\exists x$). Elle est libre sinon.
- Une variable x est libre dans F si elle a au moins une occurrence libre dans F

quelques exemples

1. $\forall x\ P(x)$ x est lié
2. $Q(x)$ x est libre
3. $\forall x\ (P(x)) \wedge Q(x)$ x une *occurence* de x est liée, l'autre est libre.

problème d'une variable à la fois libre et liée

On a dit (mais on verra pourquoi dans le chapitre sur la sémantique) que l'on peut toujours renomer une variable **liée**.
Par exemple $\forall x\ (P(x)) \wedge Q(x)$ a exactement le même sens que $\forall z\ (P(z)) \wedge Q(x)$ et dans cette dernière formule aucune variable n'est à la fois libre et liée (on parle de formule **polie** ou **propre**).

formule et sous forrmule

$$\overbrace{\exists y\ \underbrace{\forall x\ (P(x) \rightarrow R(x, y))}_{A}}^{F}$$

les deux occurences de x sont liées dans F comme A
y est lié dans F, mais libre dans A

Définition par induction de l' ensemble $VarLib$ des variables libres d'une fbf

les variables ayant une occurrence non quantifiée

- base : quand F est un atome, $VarLib(F) =_{def} Var(F)$
- induction :
  - Si $F =_{def} \neg A$, $VarLib(F) =_{def} VarLib(A)$
  - Si $F =_{def} (A \wedge B)$, $VarLib(F) =_{def} VarLib(A) \cup VarLib(B)$
  - Si $F =_{def} \forall x\ A$, $VarLib(F) =_{def} VarLib(A) - \{x\}$

Définition par induction de l' ensemble $VarLiees$ des variables liées d'une fbf

les variables ayant une occurrence quantifiée

- base : quand F est un atome, $VarLiees(F) =_{def} \emptyset$
- induction :
  - Si $F =_{def} \neg A$, $VarLiees(F) =_{def} VarLiees(A)$
  - Si $F =_{def} (A \wedge B)$, $VarLiees(F) =_{def} VarLiees(A) \cup VarLiees(B)$
  - Si $F =_{def} \forall x\ A$, $VarLiees(F) =_{def} VarLiees(A) \cup \{x\}$

fbf ouverte et fermée Une fbf est ouverte lorsqu'elle a au moins une variable libre et elle est fermée sinon (i.e si elle n'a aucune variable libre)

utilisation

- les formules ouvertes ont une sémantique qui n'est pas claire (comment les interpréter)
- les formules fermées sont celles qu'on utilise pour **représenter** des connaissances.
- les formules ouvertes servent au **calcul** (de la valeur de vérité pour une interprétation donnée), aux démonstrations et aux définitions.

2.6 Quelques exercices

(à faire)

1. contrôle 2007
   soit la formule
   $\forall w\ (P(w) \vee (\exists x\ Q(x) \wedge \exists y\ \forall z\ R(x,y,z)))$
   (a) dessiner l'arbre syntaxique de cette formule
   (b) cette formule est-elle fermée ?
2. exercice de modélisation
   On dispose des relations
   - $Pere(x,y)$ dont la sémantique est : x est le père de y
   - $Mere(x,y)$ dont la sémantique est : x est la mère de y

   (a) quelle est la sémantique du symbole de prédicat Par défini par $\forall x\ \forall y\ Par(x,y) \leftrightarrow Pere(x,y) \vee Mere(x,y)$
   (b) définir de même le symbole de prédicat $GdPar(x,y)$ dont la sémantique est : x est un grand parent de y (vous avez le droit d'utiliser le symbole de prédicat Par.
   (c) donner la formule qui signifie qu'on ne peut être son propre parent.