

Logique des prédicats : première partie

1 Introduction

1.1 Exemple introductif

La logique des propositions a une expressivité limitée (par rapport par exemple au langage naturel)

Exemple :

- $A =_{def}$ Tout étudiant possède un bac
- $B =_{def}$ Pierre est un étudiant
- $C =_{def}$ Pierre possède un bac
- Comment avoir $\{A, B\} \models C$

1.2 De la logique des propositions simples ...

1. modifier l'énoncé à modéliser :

$A_{Pierre} =_{def}$ Si Pierre est un étudiant alors Pierre possède un bac

On obtient bien $\{A_{Pierre}, B\} \models C$

2. Conséquence :

Idée 1 : Introduction d'un domaine : *l'ensemble des objets du monde considéré*

– Domaine : l'ensemble des personnes : $\{p_1, p_2, \dots \text{ Pierre } \dots\}$

– La modélisation de A devient :

- $A_1 =_{def}$ Si p_1 est un étudiant alors p_1 possède un bac
- $A_2 =_{def}$ Si p_2 est un étudiant alors p_2 possède un bac
- ...

3. Problèmes :

- Infinité potentielle du domaine
- Accroissement inutile du nombre de symboles propositionnels
- Comment représenter *il y a au moins un étudiant* ?

1.3 ... aux propositions paramétrées : les symbole propositionnel d'arité donnée ...

Idée 2 : utiliser des énoncés paramétrés par des variables dont les valeurs seront prises dans le domaine

Penons un autre exemple :

Chercons à modéliser

- François n'est pas coupable
- François est le pote à Emile
- Emile est le pote à Denis
- Denis est le pote à Charles
- Charles est le pote à Bernard
- Bernard est le pote à Albert
- le pote à un coupable est coupable
- Donc Albert n'est pas coupable

modélisation en logique des propositions

modélisation pure on doit modifier l'énoncé à modeliser

- $\neg C_F$
- $pote_{FG} \wedge pote_{EF} \wedge pote_{DE} \wedge pote_{CD} \wedge pote_{AB}$
- $A_{coup} \wedge pote_{AB} \rightarrow B_{coup}$
- $B_{coup} \wedge pote_{BC} \rightarrow C_{coup}$
- $C_{coup} \wedge pote_{CD} \rightarrow D_{coup}$
- $D_{coup} \wedge pote_{DE} \rightarrow E_{coup}$
- $E_{coup} \wedge pote_{EF} \rightarrow F_{coup}$
- $F_{coup} \wedge pote_{FG} \rightarrow G_{coup}$
- $\models \neg A_{coup}$

il y a 11 symboles propositionnels et rajouter Gérard rajoute deux symboles propositionnels.

modélisation paramétrée

- $\neg Coup(F)$
- $Pote(F, E) \wedge Pote(E, D) \wedge Pote(D, C) \wedge Pote(C, B) \wedge Pote(B, A)$
- $Coup(A) \wedge Pote(A, B) \rightarrow Coup(B)$
- $Coup(B) \wedge Pote(B, C) \rightarrow Coup(C)$
- $Coup(C) \wedge Pote(C, D) \rightarrow Coup(D)$
- $Coup(D) \wedge Pote(D, E) \rightarrow Coup(E)$
- $Coup(E) \wedge Pote(E, F) \rightarrow Coup(F)$
- $\models \neg A_{coup}$

il y a 7 symboles et rajouter Gérard rajoute un symbole propositionnel.

Mais maintenant *Coup* est un symbole propositionnel d'arité 1 et *Pote* est un symbole propositionnel d'arité 2.

1.4 ... avec des quantificateurs ...

Idée 3 : utiliser des quantificateurs précisant les valeurs qui peuvent être prises par les variables

- Tous les éléments :
 - Quantification universelle : \forall
 - Exprimant : pour tout, quelque soit, chaque ...
- Au moins un élément :
 - Quantification Existentielle : \exists
 - Exprimant : il y a, certains, quelque ...
- Exemple :
 - $A =_{def} \forall x (EtreUnEtudiant(x) \rightarrow AvoirUnBac(x))$
 - $B =_{def} EtreUnEtudiant(Pierre)$
 - $C =_{def} AvoirUnBac(Pierre)$
 - On a bien $\{A, B\} \models C$

1.5 Prédicat vs. Propositions

- La logique des propositions représente le monde par des faits
- La logique des prédicats représente le monde par
 - des objets : les éléments du domaine p_1, p_2 Pierre ...
 - des propriétés sur ces objets et relations entre ces objets : *EtreUnEtudiant*, *PosséderUnBac* ... *Pair*, *<*, *EtreLePgcdDe* ...Ces propriétés et ces relations sont appelées des *prédicats* car ce sont des fonctions dont l'image est un booléen.
- et des fonctions entre ces objets : *PèreDe* ... *Succ*, *Carré*, *+* ... (on n'étudiera pas cet aspect dans la première partie)

2 Syntaxe de la logique des prédicats

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.

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 langage du premier ordre

C'est un ensemble $\mathcal{L} = Cst \cup Prd$ de symboles constitué :

- d'un ensemble Cst de constantes (ou symboles de fonctions 0-aires) : $\{a, b, c \dots\}$
 Dans le cadre de cette première partie nous ne considérons pas les symboles de fonctions d'arité ≥ 1 : $\{f, g, h \dots\}$
- d'un ensemble Prd de symboles de prédicats avec une arité associée : $\{P_1, Q_2, R_2 \dots\}$
 Les symboles de prédicats d'arité 0 sont les symboles de proposition de la logique des propositions
- $Cst \cap Prd = \emptyset$

2.2 Termes

Un terme est une expression désignant un objet du domaine modélisé

En plus des constantes, on dispose d'un ensemble infini V de variables disjoint de $\mathcal{L} : \{x, y, z \dots\}$ et de façon optionnelle de $\{\top, \perp\}$, deux symboles spéciaux (qui ne sont ni dans V ni dans \mathcal{L}).

L'ensemble des termes d'un langage $\mathcal{L} =_{def} Cst \cup Prd$ est l'ensemble $T_m =_{def} V \cup Cst$

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 $\mathcal{L} =_{def} Cst \cup Prd$ un langage, V un ensemble infini de variables, $C =_{def} \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ les connecteurs, $Q =_{def} \{\forall, \exists\}$ les quantificateurs, et $D =_{def} \{(\, , \,)\}$ 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
 $\{ p(t_1, \dots, t_n) \mid p \in Prd \text{ est un prédicat } n\text{-aire et } t_1, \dots, t_n \in T_m \text{ sont des termes} \}$
 2. $FBF(\mathcal{L})$ contient $\{\top, \perp\}$ dans la mesure où ces symboles sont admis.
 3. **le cas spécial du symbole d'égalité** =
 $FBF(\mathcal{L})$ contient l'ensemble des formules $\{t_1 = t_2 \text{ avec } t_1 \text{ et } t_2 \in T_m =_{def} Cst \cup V\}$
- induction : soit A et $B \in FBF(\mathcal{L})$ et soit $x \in V$:
 - $\neg A \in FBF(\mathcal{L})$
 - $(A \wedge B) \in FBF(\mathcal{L})$ (idem avec $(A \vee B), (A \rightarrow B), (A \leftrightarrow B)$)
 - $\forall x A \in FBF(\mathcal{L})$ (idem avec $\exists x A$)

2.4 Définitions par induction

- Arbre syntaxique d'une fbf – les feuilles sont étiquetées par des atomes
- Profondeur d'une fbf
- Ensemble des sous-fbf d'une fbf
- Ensemble $Var(F)$ des variables d'une fbf F
 - base
 - Si F est un atome de la forme $F =_{def} p(t_1, \dots, t_n)$, $Var(F) =_{def} \{t_1, \dots, t_n\} \cap V$
 - Si $F =_{def} \top \text{ ou } \perp$ alors $Var(F) =_{def} \emptyset$
 - induction
 - Si $F =_{def} \neg A$, $Var(F) =_{def} Var(A)$
 - Si $F =_{def} (A \wedge B)$, $Var(F) =_{def} Var(A) \cup Var(B)$ (id. $\vee, \rightarrow, \leftrightarrow$)
 - Si $F =_{def} \forall x A$, $Var(F) =_{def} Var(A) \cup \{x\}$ (id. $\exists x$)

2.5 Variables libres et liées

Nous voulons parler des différences et ressemblance entre les trois formules

$$\begin{aligned} & \forall x \exists y P(x, y) \\ & \forall x P(x, y) \\ & (\forall x \exists y P(x, y)) \wedge (\forall x P(x, y)) \end{aligned}$$

- 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 $F =_{def} \forall x A$, $VarLib(F) =_{def} VarLib(A) - \{x\}$
- Ensemble $VarLies$ des variables liées d'une fbf (les variables ayant une occurrence quantifiée)
 - base : quand F est un atome, $VarLies(F) =_{def} \emptyset$
 - induction :
 - Si $F =_{def} \neg A$, $VarLies(F) =_{def} VarLies(A)$
 - Si $F =_{def} (A \wedge B)$, $VarLies(F) =_{def} VarLies(A) \cup VarLies(B)$
 - Si $F =_{def} \forall x A$, $VarLies(F) =_{def} VarLies(A) \cup \{x\}$

2.6 Portée d'un quantificateur

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

2.7 Fbf ouverte et fermée

Dans une fbf une même variable peut être à la fois libre et liée il est parfois nécessaire de préciser de quelle occurrence d'une variable on parle

- 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

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

2.8 Egalité syntaxique à un renommage près

Le nom des variables liées n'est pas important

$$\forall x (P(x) \rightarrow Q(x, y)) \equiv^1 \forall z (P(z) \rightarrow Q(z, y))$$

On dit que deux formules sont α -équivalentes si et seulement si elles sont syntaxiquement identiques à un renommage près des occurrences liées des variables

Attention, on ne doit pas en renommant transformer une occurrence libre en occurrence liée

$$\forall x (P(x) \rightarrow Q(x, y)) \equiv \forall z (P(z) \rightarrow Q(z, y)) \not\equiv \forall y (P(y) \rightarrow Q(y, y))$$

1. on verra plus loin ce que contient ce signe \equiv

3 Sémantique de la logique des prédicats

Une *interprétation* I d'un langage \mathcal{L} est un couple $I =_{def} (D, i)$ tel que :

- D (le domaine d'interprétation) est un ensemble non vide
- pour toute constante a de \mathcal{L} , $i(a)$ est un élément de D
- pour tout prédicat P de \mathcal{L} , $i(P)$ est une application de $D^{arit(P)}$ dans $\{faux, vrai\}$

Exemples pour $\mathcal{L} =_{def} \{a, b, P_1, Q_2\}$ et $I =_{def} (D, i)$ avec $D =_{def} \{d\}$ on aura
 $i(a) =_{def} d$, $i(b) =_{def} d$, $i(P_1) : D \rightarrow \{faux, vrai\}$ $i(Q_2) : D \times D \rightarrow \{faux, vrai\}$
 et on peut alors avoir pour i les deux interprétations

- $i_1(P) =_{def} \{(d, vrai)\}$ et $i_1(Q) =_{def} \{((d, d), vrai)\}$
- $i_2(P) =_{def} \{(d, faux)\}$ et $i_2(Q) =_{def} \{((d, d), vrai)\}$

ou bien on peut aussi définir l'interprétation

$D =_{def} \mathbb{N}$, $i_3(a) =_{def} 0$, $i_3(b) =_{def} 3$, $i_3(P) =_{def} pair?$ et $i_3(Q) =_{def} <$

3.1 Valeur d'une fbf

- I étant une interprétation d'un langage \mathcal{L} nous voulons définir une fonction $v(F, I)$ dans $\{faux, vrai\}$ qui associera à toute fbf fermée F construite sur \mathcal{L} une valeur de vérité.
- Ceci se fait en utilisant l'interprétation I , et le sens donné en logique des propositions aux connecteurs
- Non seulement il faut également interpréter les quantificateurs, mais surtout la sémantique compositionnelle nécessite d'interpréter des fbf ayant des variables libres (même si les fbf considérées à l'origine sont fermées). On fait cela en utilisant la notion d'assignation des variables

assignation

Une *assignation* est une application de l'ensemble des variables dans le domaine d'interprétation D .

exemple

pour le domaine $D =_{def} \mathbb{N}$, on peut définir les assignations de x et y

- $\sigma_1(x) =_{def} 3$ $\sigma_1(y) =_{def} 2$
- $\sigma_2(x) =_{def} 0$ $\sigma_2(y) =_{def} 1$
- $\sigma_3(x) =_{def} 2$ $\sigma_3(y) =_{def} 1$

valeur d'une fbf pour cet exemple

On veut avoir

- $v(Q(x, y), i_3, \sigma_1) = faux$
- $v(Q(x, y), i_3, \sigma_2) = vrai$
- $v(Q(x, y), i_3, \sigma_3) = faux$
- $v(a = x, i_3, \sigma_2) = vrai$
- $v(a = x, i_3, \sigma_3) = faux$
- $v(\neg P(y) \rightarrow Q(x, y), i_3, \sigma_1) = vrai$
- $v(\neg P(y) \rightarrow Q(x, y), i_3, \sigma_2) = vrai$
- $v(\neg P(y) \rightarrow Q(x, y), i_3, \sigma_3) = faux$
- $v(\forall y \neg P(y) \rightarrow Q(x, y), i_3, \sigma_1) = faux$: si x est assigné à 3, il suffit de prendre l'assignation de la variable **liée** y à 1
- $v(\forall y \neg P(y) \rightarrow Q(x, y), i_3, \sigma_2) = vrai$

Définition par induction de la valeur d'une fbf

Soit $I =_{def} (D, i)$ une interprétation d'un langage \mathcal{L} et σ une assignation de V dans D , $v(F, I, \sigma)$ est telle que :

- base
 1. si $F =_{def} P(t_1, t_2, \dots, t_n)$ est un atome, $v(F, I, \sigma) =_{def} i(P)(\phi(t_1, I, \sigma), \dots, \phi(t_n, I, \sigma))$, où :
 - si t est une constante $\phi(t, I, \sigma) =_{def} i(t)$
 - si t est une variable $\phi(t, I, \sigma) =_{def} \sigma(t)$
 2. $v(\perp, I, \sigma) =_{def} faux$
 3. $v(\top, I, \sigma) =_{def} vrai$
 4. **le cas spécial du symbole d'égalité** =
 $v(t_1 = t_2, I, \sigma)$ est défini de quatre façons, suivant que t_1 et t_2 sont des variables ou des constantes : il s'agit de vérifier l'identité de deux objets O_1 et O_2 dans D , ce que l'on notera $O_1 =_D O_2$.
 - $t_1 \in Cst$ et $t_2 \in Cst$ $v(t_1 = t_2, I, \sigma)$ vaut vrai si et seulement si $I(t_1) =_D I(t_2)$
 - $t_1 \in Cst$ et $t_2 \in V$ $v(t_1 = t_2, I, \sigma)$ vaut vrai si et seulement si $I(t_1) =_D \sigma(t_2)$
 - $t_1 \in V$ et $t_2 \in Cst$ $v(t_1 = t_2, I, \sigma)$ vaut vrai si et seulement si $\sigma(t_1) =_D I(t_2)$
 - $t_1 \in V$ et $t_2 \in V$ $v(t_1 = t_2, I, \sigma)$ vaut vrai si et seulement si $\sigma(t_1) =_D \sigma(t_2)$
- induction
 - si $F =_{def} \neg A$, alors $v(F, I, \sigma) =_{def} NON(v(A, I, \sigma))$
 - si $F =_{def} (A \wedge B)$, $v(F, I, \sigma) =_{def} ET(v(A, I, \sigma), v(B, I, \sigma))$ (id. avec $\vee \rightarrow \leftrightarrow$)
 - si $F =_{def} \forall x A$, $v(F, I, \sigma) =_{def} vrai$ si et seulement si pour tout élément d de D , $v(A, I, \sigma_{x \triangleright d}) = vrai$ où $\sigma_{x \triangleright d}$ est l'assignation obtenue à partir de σ en donnant en plus à la variable x la valeur d
 - si $F =_{def} \exists x A$, $v(F, I, \sigma) =_{def} vrai$ si et seulement si il existe un élément d de D t.q. $v(A, I, \sigma_{x \triangleright d}) = vrai$

3.2 Propriétés

- Pour toute assignation σ , $v(F, I, \sigma)$ ne dépend que de la restriction de σ à l'ensemble des variables libres de F
- Pour une fbf fermée F la valeur de $v(F, I, \sigma)$ est indépendante de σ et on la note $v(F, I)$.
- Si F n'est pas fermé, et en appelant $x_1, x_2 \dots x_n$ l'ensemble des variables libres de F
 - $v(\exists x_1 \exists x_2 \dots \exists x_n F, I)$ vaut vrai si et seulement si il existe une assignation σ telle que $v(F, I, \sigma) = vrai$
 - $v(\forall x_1 \forall x_2 \dots \forall x_n F, I)$ vaut vrai si et seulement si pour toute assignation σ on a $v(F, I, \sigma) = vrai$

3.3 Définitions

Modèle et contre-modèle

- Une interprétation I t.q. il existe une assignation σ avec $v(F, I, \sigma) = vrai$ est appelée un modèle de F
- Une interprétation I t.q. il existe une assignation σ avec $v(F, I, \sigma) = faux$ est appelée un contre-modèle de F

Soit F une fbf :

- F est satisfiable si elle possède au moins un modèle
- F est contingente si elle possède au moins un modèle et au moins un contre-modèle
- F est insatisfiable si elle ne possède aucun modèle
- F est valide si toute interprétation est un modèle

Propriétés :

- F est valide si et seulement si $\neg F$ est insatisfiable
- F est contingente si et seulement si $\neg F$ est contingente
- F est insatisfiable si et seulement si $\neg F$ est valide

3.4 Equivalence

Définition Deux fbf A et B (construites sur un même langage du premier ordre \mathcal{L}) sont dites logiquement équivalentes, $A \equiv B$, lorsque pour toute interprétation I et toute assignation σ : $v(A, I, \sigma) = v(B, I, \sigma)$

Propriété $A \equiv B$ si et seulement si $A \leftrightarrow B$ est valide.

Il faut connaitre aussi les formules $\neg(\exists x p(x)) \equiv \forall x \neg p(x)$ et $\neg(\forall x p(x)) \equiv \exists x \neg p(x)$

3.5 Conséquence logique

Si H_1, H_2, \dots, H_n et C sont des fbf fermées d'un langage \mathcal{L} du premier ordre on dit que C est conséquence logique de H_1, H_2, \dots, H_n , lorsque toute interprétation I de \mathcal{L} qui est un modèle de tous les H_i est un modèle de C : $\{H_1, H_2, \dots, H_n\} \models C$.

3.5.1 Equivalence et conséquence logique

si F_1 et F_2 sont des fbf fermées d'un langage \mathcal{L} du premier ordre, on a immédiatement $F_1 \equiv F_2$ si et seulement si $F_1 \models F_2$ et $F_2 \models F_1$

3.6 Théorème

Si H_1, H_2, \dots, H_n et C sont des fbf fermées d'un langage du premier ordre les propriétés ci-dessous sont équivalentes :

1. $\{H_1, H_2, \dots, H_n\} \models C$
2. $H_1 \wedge H_2 \wedge \dots \wedge H_n \rightarrow C$ est valide
3. $H_1 \wedge H_2 \wedge \dots \wedge H_n \wedge \neg C$ est insatisfiable

On peut maintenant parler de la "validité", ou "correction", d'un "raisonnement". De façon générale, un raisonnement consiste à obtenir une conclusion à partir d'un ensemble d'hypothèses. Imaginons que l'on formule les hypothèses $H_1, H_2 \dots H_N$ (chacune étant une fbf) et qu'on en conclut C (une fbf). Ce raisonnement est correct (ou valide) si C est une conséquence logique de $H_1, H_2 \dots H_N$.

4 Démonstrations d'équivalence

Pour montrer que deux formules F_1 et F_2 sont équivalentes, on peut s'appuyer directement sur la notion d'interprétation ; on peut montrer que toute interprétation qui rend vraie F_1 rend aussi vraie F_2 , et que toute interprétation qui rend fausse F_1 rend aussi fausse F_2 (on applique la définition de l'équivalence) ; ou bien, on peut montrer que tout modèle de F_1 est un modèle de F_2 (c'est-à-dire que $F_1 \models F_2$), et réciproquement (c'est-à-dire alors que $F_1 \equiv F_2$). Nous étudions ci-dessous d'autres façons de montrer l'équivalence entre deux formules : en s'appuyant sur des équivalences connues pour la logique des propositions (voir *théorème des tautologies*) ou en passant d'une formule à l'autre en remplaçant des sous-formules par des formules équivalentes (voir *théorème de substitution*).

4.1 Théorème des tautologies

- Soit F une fbf valide de la logique des **propositions** ayant p_1, p_2, \dots, p_n pour symboles propositionnels.
- Soient F_1, F_2, \dots, F_n des fbf (pas nécessairement fermées) d'un langage du premier ordre \mathcal{L}
- Alors F' la fbf obtenue à partir de F en substituant, pour tout i , F_i à p_i , est valide.

Utilisation

Si F_1, F_2 et F_3 sont des fbf d'un langage du premier ordre on a, par exemple, les équivalences :

- $\neg\neg F_1 \equiv F_1$
- $(F_1 \wedge F_2) \equiv (F_2 \wedge F_1)$
- $F_1 \wedge (F_2 \wedge F_3) \equiv (F_1 \wedge F_2) \wedge F_3$
- $\neg(F_1 \wedge F_2) \equiv \neg F_1 \vee \neg F_2$
- $(F_1 \rightarrow F_2) \equiv \neg F_1 \vee F_2$

Attention toute formule valide au premier ordre ne peut pas provenir par substitution d'une formule valide des propositions : $(\neg\exists x p(x) \rightarrow \forall x \neg p(x))$

4.2 Substitution de variables

Définition On appelle substitution de variables l'opération qui consiste à remplacer dans une formule F toutes les occurrences libres d'une variable x par un terme t

On note $\{x \leftarrow t\}F$ le résultat de la substitution de x par t dans F .

Exemple $\{x \leftarrow a\}(P(x) \rightarrow (\forall x Q(x, y) \vee \exists z Q(x, z))) = P(a) \rightarrow (\forall x Q(x, y) \vee \exists z Q(a, z))$

4.3 Théorème de substitution

Le théorème de substitution permet d'obtenir, par substitution d'une fbf à une fbf équivalente, des équivalences entre fbf du premier ordre à partir d'équivalences entre fbf du premier ordre : Soient F une fbf, S une sous-fbf de F , et S' une fbf équivalente à S . Toute fbf F' obtenue à partir de F en remplaçant une occurrence de S par une occurrence de S' est équivalente à F .

Nous allons donner quelques équivalences qui ne peuvent pas être obtenues avec le théorème de tautologie car elles concernent des remplacements ou déplacements de quantificateurs. Ci-dessous, A et B désignent des fbf, et l'on ne fait pas d'hypothèse sur les variables qu'elles contiennent.

- si $y \notin \text{var}(A)$: $Qx A \equiv Qy \{x \leftarrow y\} A$ où $Q \in \{\forall, \exists\}$ *renommage d'une variable liée*
- si $x \notin \text{var}(A)$: $\forall x A \equiv \exists x A \equiv A$
- si $x \notin \text{var}(B)$: $(Qx A) op B \equiv Qx (A op B)$ où $Q \in \{\forall, \exists\}$ et $op \in \{\wedge, \vee\}$
- $\neg \forall x A \equiv \exists x \neg A$
- $\neg \exists x A \equiv \forall x \neg A$
- $\forall x A \wedge \forall x B \equiv \forall x (A \wedge B)$ *distributivité du \forall par rapport au \wedge*
- $\exists x A \vee \exists x B \equiv \exists x (A \vee B)$ *distributivité du \exists par rapport au \vee*
- si $y \notin \{\text{var}(A) \cup \text{var}(B)\}$: $Qx A \wedge Q'y B \equiv Qx Q'y (A \wedge \{x \leftarrow y\} B)$ où Q et $Q' \in \{\forall, \exists\}$
- si $y \notin \{\text{var}(A) \cup \text{var}(B)\}$: $Qx A \vee Q'y B \equiv Qx Q'y (A \vee \{x \leftarrow y\} B)$ où Q et $Q' \in \{\forall, \exists\}$

exemple de preuve : pour $\neg \forall x A \equiv \exists x \neg A$
on va examiner deux cas :

- le cas général : $\forall x A$ est une formule ouverte (donc $\exists x \neg A$ aussi)
Il faut introduire la notion d'assignation dès le départ : soit I une interprétation et σ une assignation, $v(\neg \forall x A, I, \sigma)$ vaut vrai si et seulement si $v(\forall x A, I, \sigma)$ vaut faux ce qui revient à dire si et seulement si il existe un élément $d \in D$ tel que $v(A, I, \sigma \cup \{x \leftarrow d\})$ vaut faux ou, ce qui revient au même, si et seulement si il existe un élément $d \in D$ tel que $v(\neg A, I, \sigma \cup \{x \leftarrow d\})$ vaut vrai c'est à dire si et seulement si $v(\exists x \neg A, I, \sigma)$ vaut vrai
- dans le cas particulier où $\forall x A$ est une formule fermée (donc $\exists x \neg A$ aussi), la preuve est essentiellement la même :
une interprétation I sur un domaine D est un modèle de $\neg \forall x A$ si et seulement si I n'est pas un modèle de $\forall x A$, c'est à dire si et seulement si I n'est pas, pour tout élément $d \in D$, un modèle de $\{x \leftarrow d\}A$, autrement dit si et seulement si il existe un élément $d \in D$ tel que I ne soit pas un modèle de $\{x \leftarrow d\}A$, ce qui revient à dire si et seulement si il existe un élément $d \in D$ tel que I soit un modèle de $\{x \leftarrow d\} \neg A$, autrement dit si et seulement si I est un modèle de $\exists x \neg A$.