Logique des prédicats et théories du 1er ordre

Aide mémoire

G. Falquet, 1999-2002

Table des matières

1 Introduction

La logique des prédicats a pour but de généraliser la logique des propositions. On peut considérer un prédicat comme un énoncé général où apparaissent des variables. Par exemple:

“X est la soeur de Y”

“si X est le père de Y et Y le père de Z alors X est le grand-père de Z”

Si l’on remplace toutes les variables d’un prédicat par des valeurs définies on obtient une proposition à laquelle on pourra associer une interprétation (vrai, faux). Par exemple:

X = Anne et Y = Eugène dans (1) donne: “Anne est la soeur de Eugène”.

Un prédicat repésente donc potentiellement une classe de propositions.
Par l’introduction de quantificateurs on peut représenter le fait qu’un énoncé est vrai pour toutes les valeurs possibles des variables ou qu’il existe au moins une valeur des variables qui rend l’énoncé vrai. Par exemple: “quel que soit X si X est un homme alors X est mortel”. Les variables pouvant prendre leur valeur dans des ensembles infinis, les quantificateurs permettent donc d’énoncer des faits correspondant à une infinité de propositions.

2 Le langage

2.1 Syntaxe

Pour écrire des formules de logique des prédicats, on commence par se donner un vocabulaire W composé de symboles de différents types:

- variables (x, y, z, x1, y1, z1, …)
- constantes (a, b, c, a1, b1, c1, …)
- fonctions (f, g, h, f1, g1, h1, …)
- prédicats (P, Q, R, P1, Q1, R1, …)
- parenthèses
- connecteurs logiques: $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$
- quantificateurs: $\exists$, $\forall$

à chaque symbole de fonction et de prédicat est associée une arité qui est un entier positif ou nul. On peut considérer que les constantes sont des fonctions 0-aires.
Un terme du langage peut être:

- un symbole de constante
- un symbole de variable
- $f(t_1, \ldots, t_n)$ où f est un symbole de fonction n-aire et $t_1, \ldots, t_n$ sont des termes.

Un atome est un énoncé de la forme

- $P(t_1, \ldots, t_k)$ où P est un symbole de prédicat k-aire et $t_1, \ldots, t_k$ sont des termes.

Un littéral est soit un atome, soit un atome précédé du symbole de négation $\neg$.
Finalement une formule peut être:

- un littéral
- une quantification universelle $(\forall x.w)$ où x est une variable et w est une formule
- une quantification existentielle $(\exists x.w)$ où x est une variable et w est une formule
- $\neg(w)$, $(w1) \vee (w2)$, $(w1) \wedge (w2)$, $(w1) \Rightarrow (w2)$, $(w1) \Leftrightarrow (w2)$, où w, w1 et w2 sont des formules.

2.2 Variables libres et liées

Les variables qui apparaissent dans une formule sont dites libres ou liées, selon le principe suivant:

- toutes les variables d'une formule sans quantificateurs sont libres
- si x est libre dans w, x est liée dans $\forall x.w$ et dans $\exists x.w$

Une formule dont toutes les variables sont liées est dite fermée. Par exemple:

$$\forall x.P(x) \Rightarrow (\exists y.Q(x, y))$$

Si ce n'est pas le cas, la formule est dite ouverte. Pour une formule ouverte w dont les variables x, y, et z sont libres on notera w(x, y, z) lorsqu'on désirera indiquer quelles sont les variables libres de w.

$$w(x, z) = P(x) \Rightarrow (\exists y.Q(x, y, z))$$

est une formule ouverte où x et z sont libres.

2.3 Substitution et instantiation

L'opération de substitution consiste à remplacer (purement syntaxiquement) certaines variables libres d'une formule w par des termes.

Si w est une formule où $x_1, \ldots, x_n$ sont des variables libres, et σ est la substitution $(t_1/x_1, \ldots, t_n/x_n)$, $w\sigma$ dénote la formule obtenue en remplaçant toutes les occurences de x_i dans w par t_i (pour i = 1 à n).

Exemples.

$w = Q(x_1, y_1) \Leftrightarrow \forall x_2.(R(x_2, z_1) \vee S(x_2, y_1))$

$\sigma = (z_8/x_1, g(a, b)/y_1)$

$w\sigma = Q(z_8, g(a, b))\ ) \Leftrightarrow \forall x_2.(R(x_2, z_1) \vee S(x_2, g(a, b)))$

$w(c_{23}/z_1) = Q(x_1, y_1)\} \Leftrightarrow \forall x_2.(R(x_2, c_{23}) \vee S(x_2, y_1))$

On dira qu'une substitution instancie x si elle remplace x par un terme où n'apparaît aucune variable. La substitution (c_{23}/z_1) de l'exemple précédent instancie z_1 alors que la substitution $(z_8/x_1, g(a, b)/y_1)$ instancie y_1 mais pas x_1.

3 Sémantique du calcul des prédicats

Le sens d'une formule fermée sera l'une des valeurs t ou f de l'anneau booléen. Cette valeur sera calculée à partir des interprétations des constantes, fonctions et prédicats et par application des règles propres aux connecteurs logiques et aux quantificateurs. La présence de variables nécessite l'introduction d'opérations de substitutions.

3.1 Interprétations

Pour interpréter une formule construite sur un vocabulaire W on commence par se donner une interprétation de chaque symbole de W.

Une interprétation I d'un vocabulaire W est la donnée de:

- un ensemble D appelé *domaine* de l'interprétation
- pour chaque constante c de W, un élément c_I de D
- pour chaque symbole de fonction n-aire f, une fonction f_I de D^n dans D
- pour chaque symbole de prédicat n-aire P, une relation n-aire P_I sur D (i.e. un sous-ensemble de D^n)

Pour calculer la valeur d'un terme ou d'une formule on affectera à ses variables libres des valeurs prises dans le domaine D
La valeur d'un terme t pour une interprétation I et une affectation $\alpha = [x_1 \leftarrow d_1, \ldots, x_n \leftarrow d_n]$, que l'on notera $t^I\alpha$ est définie par les règles suivantes:

constantes: si t = c, $t^I\alpha = c_I$

- c'est simplement la valeur du domaine D qui correspond à C

variables: si $t = x_i$ alors $t^I\alpha = d_I$

- c'est la valeur affectée à x_i par α)

fonctions: si $t = f(z_1, \ldots, z_p)$ alors $t^I\alpha = f_I(z_1{}^I\alpha, \ldots, z_p{}^I\alpha)$

- on commence par calculer la valeur des termes qui servent de paramètres à la fonction, puis on applique la fonction f_I sur ces valeurs.

Exemple 1

On considère le vocabulaire W = {c, d, e, x, y, f (unaire), g (binaire)}

Soit D = {1, 2, 4} et l'interprétation I

- $c_I = 2$; $d_I = 1$; $e_I = 4$;
- $f_I = \{1 \to 1; 2 \to 1; 4 \to 2\}$;
- $g_I = \{(1,1) \to 1; (1,2) \to 2; (1,4) \to 4; (\_, \_) \to 1$ pour les autres valeurs de paramètres$\}$

1. $f(e)^I = f_I(e_I) = f_I(4) = 2.$
2. $x^I[x \leftarrow 2, y \leftarrow 1] = 2.$
3. $f(x)^I[x \leftarrow 2] = f_I(x^I[x \leftarrow 2]) = f_I(2) = 1.$
4. $g(x, f(y))^I[x \leftarrow 1, y \leftarrow 4] = g_I(x^I[x \leftarrow 1, y \leftarrow 4], f(y)^I[x \leftarrow 1, y \leftarrow 4])$
   $= g_I(1, f(y)^I[x \leftarrow 1, y \leftarrow 4]) = g_I(1, f_I(y^I[x \leftarrow 1, y \leftarrow 4]))$
   $= g_I(1, f_I(4)) = g_I(1,2) = 2.$

La valeur d'une formule $w(x_1, \ldots, x_k)$ pour une interpretation I et une affectation α , notée $w^I\alpha$ est définie par les règles suivantes:

Formules atomiques

Si w est une formule atomique $P(t_1, \ldots, t_n)$,
$w^I\alpha$ est vraie si le n-tuple $\langle t_1{}^I\alpha, \ldots, t_n{}^I\alpha \rangle \in P_I$, elle est fausse sinon

Formules composées

On applique les règles habituelles des connecteur logiques :

- $(\neg w)^I\alpha$ = vrai si $(w)^I\alpha$ = faux
  = faux sinon
- $(w_1 \vee w_2)^I\alpha$ = faux si $w_1{}^I\alpha = w_2{}^I\alpha$
  = vrai sinon
- $(w_1 \wedge w_2)^I\alpha$ = vrai si $w_1{}^I\alpha = w_2{}^I\alpha$ = vrai,
  = faux sinon
- $(w_1 \Rightarrow w_2)^I\alpha$ = faux si $w_1{}^I\alpha$ = vrai et $w_2{}^I\alpha$ = faux,
  = vrai sinon
- $(w_1 \Leftrightarrow w_2)^I\alpha$ = vrai si $w_1{}^I\alpha = w_2{}^I\alpha$,
  = faux sinon

Formules quantifiées

- $(\forall x.w)^I\alpha$ = vrai si pour tout élément $d \in D$ $w^I[\alpha, x \leftarrow d] = v$,
  = faux sinon

La notation $[\alpha, x \leftarrow d]$ désigne l'affectation formée de α, à laquelle on ajoute $x \leftarrow d$.

Une formule quantifiée universellement est donc vraie si pour toute affectation d'une valeur du domaine à x, la formule w est vraie.

- $(\exists x.w)^I\alpha$ = vrai s'il existe un élément $d \in D$ tel que $w^I[\alpha, x \leftarrow d] = v$,
  = faux sinon

Exemple 2

Soit le vocabulaire W = {cp, ca, cmi, ct, cma (constantes), x, y (variables), Ville (1-aire), Personne (1-aire), Habite (2-aire), }

et I l'interprétation suivante définie par

D = {"Paul", "Albert", "Milan", "Tokyo", "Madrid"}

cp_I = "Paul", ca_I = "Albert", cmi_I = "Milan", ct_I = "Tokyo", cma_I = "Madrid"

$Ville_I$ = {<"Milan">, <"Tokyo">, <"Madrid">}

$Personne_I$ = {<"Paul">, <"Albert">}

$Habite_I$ = {<"Paul", "Milan">, <"Albert", "Tokyo">}.

Interprétons quelques termes, atomes et formules selon I:

1. ca^I = "Albert"
2. y^I[y ←"Milan"] = "Milan"
3. $Ville^I(ct) = <ct^I> \in Ville_I$ = <"Tokyo"> $\in Ville_I$ = vrai
4. $Habite^I(cp, ct)$ = <"Paul", "Tokyo"> $\in Habite_I$ = faux
5. $\exists x.Habite(x, ct)^I$ = vrai

   car

   $Habite(x, ct)^I$[x ← "Albert"]

   = $<x^I$[x ← "Albert"], ct^I[x ← "Albert"]> $\in Habite_I$

   = <"Albert", "Tokyo"> $\in Habite_I$

   = vrai
6. $(\forall x.\exists y.Habite(x, y))^I$ = faux

   car

   $Habite(x, y)^I$[x ← "Tokyo", y ← "Tokyo"]

   = $<x^I$[x ←"Tokyo", y ←"Tokyo"], y^I[x ←"Tokyo", y ←"Tokyo"]> $\in Habite_I$

   = <"Tokyo","Tokyo"> $\in Habite_I$

   = faux
7. $(\forall x.Personne(x) \Rightarrow \exists y,Habite(x, y))^I$ = vrai.

$Personne(x)^I$ n'est vraie que pour x = "Paul" ou "Albert", il suffit donc de vérifier que $\exists y, Habite(x, y)^I$ est vrai pour ces deux valeurs de x. C'est bien le cas. Pour x = "Paul" on prend y = "Milan" et pour x = "Albert" on prend y = "Tokyo".

3.2 Equivalence sémantique, modèles et conséquence logique

Definition 1 (Equivalence) Deux termes w_1 et w_2 sont *équivalentes* si pour toute interprétation I on a

$$w_1^I = w_2^I.$$

On notera $w_1 \approx w_2$

Par exemple: $\forall x.Personne(x) \approx \neg \exists x.(\neg Personne(x))$.

Definition 2 (Modèle) Soit $F = \{w_1, \ldots, w_n\}$ un ensemble de formules fermées, un *modèle* de F est une interprétation I telle que $w_1^I = v, \ldots, w_n^I = v$.

Definition 3 (Satisfaisabilité) Un ensemble F de formules fermées est dit *satisfaisable* s'il existe au moins un modèle de F.

S'il n'existe pas de modèle de F on dit que F est *inconsistant*.

Par exemple, l'ensemble

$$F = \{P(a, b), \neg \exists y.P(a, y)\}$$

est inconsistant.

En effet, pour que $P(a, b)^I$ soit vrai il faut que $<a_I, b_I>$ appartienne à P_I mais alors $\neg \exists y.P(a, y)^I$ est forcément fausse (prendre $y = b_I$).

Definition 4 (Conséquence logique) Soit F un ensemble de formules fermées et w une formule fermée. On dira que w est une conséquence logique de F, noté $F \models w$, si

tout modèle de F est aussi un modèle de $F \cup \{w\}$.

Si l'on considère qu'une interprétation est un monde possible, la définition ci-dessus nous dit que dans tous les mondes où les formules de F sont vraies, w est nécessairement vraie.

Il est facile de voir que si $F \models$ faux (la formule qui est toujours fausse) alors F est inconsistant.

Pour prouver que w est conséquence logique de$\{w_1, \ldots, w_n\}$ on utillisera en général le *principe de déduction* qui peut s'énoncer comme suit:

$\{w_1, \ldots, w_n\} \models w$ si et seulement si$\{w_1, \ldots, w_n, \neg w\} \models$ faux

Definition 5 On appelera *tautologie* une formule w qui est vraie pour n'importe quelle interprétation. On notera: $\models w$. On parle également de formule *valide*.

4 Théorie du premier ordre

Jusqu'à présent nous avons vu le langage de la logique des prédicats et son interprétation. Nous avons donc abordé l'aspect sémantique. Retournons maintenant à l'aspect syntaxique. Pour définir le calcul des prédicats en tant que théorie du premier ordre il nous faut ajouter au langage les axiomes et les règles d'inférence.

Axiomes

Nous avons tout d'abord les axiomes correspondant à la logique des prédicats :

1. $v \Rightarrow (w \Rightarrow v)$
2. $(v \Rightarrow (w \Rightarrow u)) \Rightarrow ((v \Rightarrow w) \Rightarrow (v \Rightarrow u))$
3. $(\neg w \Rightarrow \neg v) \Rightarrow ((\neg w \Rightarrow v) \Rightarrow w)$

puis les axiomes propres à la logique du 1er ordre

4. $\forall x.w \Rightarrow w(t/x)$ où t est un terme qui est "librement substituable pour x dans w"
5. $(\forall x.(v \Rightarrow w)) \Rightarrow (v \Rightarrow \forall x.w)$ si v ne contient aucun occurence libre de x.

On dit que le terme t est librement substituable pour x dans w si et seulement si aucune occurence libre de x dans w n'est dans une sous-formule dépendant d'un quantificateur $\forall y$ ou $\exists y$, y étant une variable qui apparaît dans t.

Par exemple, si w est la formule $\exists y.(P(y) \wedge Q(x))$, on refusera de substituer à x le terme 'y' ou le terme 'f(x, y)', à cause du $\exists y$ de w.

Par contre on pourra substituer y à x dans la formule $(\exists y.(P(y)) \wedge Q(x)$.

Afin d'éviter de construire de formules difficiles à lire on évitera en général de substituer à x un terme t qui contient l'une des variables apparaissant dans w.

Règles d'inférence

Finalement les deux règles d'inférence de la théorie du premier ordre sont:

1. Modus ponens (MP): $\{v, v \Rightarrow w\} \vdash w$
2. Généralisation (GEN): $v \vdash \forall x.v$, x étant n'importe quelle variable apparaissant ou non dans v.

Theorem 1 La théorie définie ci-dessus est

- valide (sound), tout théorème déduit à l'aide des règles est une tautologie (valide);
- consistante, il n'y a pas de formule w telle que $\vdash w$ et $\vdash \neg w$;
- complète, toute tautologie est un théorème.

On a également l'équivalent du théorème de déduction de la logique des propositions.

Theorem 2

1. Si une preuve $G, v \vdash w$ n'implique aucune application de la règle de généralisation dans laquelle la variable x est libre dans la formule de départ alors

$$G \vdash (v \Rightarrow w)$$

La restriction de généralisation est nécessaire car si la règle de généralisation nous autorise à écrire $A(x) \vdash \forall x.A(x)$, on ne peut pas en conclure $\vdash A(x) \Rightarrow \forall x.A(x)$.

2. Si $G, v \vdash w$ et v est une formule fermée alors $G \vdash (v \Rightarrow w)$.

À partir du calcul des prédicats du premier ordre on peut construire d'autres théories du premier ordre en ajoutant de nouveaux axiomes, dits axiomes propres.

Exemple 3 Théorie des ordres partiels

Axiomes propres

OP1. $\forall x.\neg (x < x)$ (antiréflexivité)

OP2. $\forall x \forall y \forall z.(x < y \wedge y < z) \Rightarrow x < z$ (transitivité)

Definition 6 Un modèle d'une théorie est une interprétation dans laquelle tous les axiomes sont vrais.

A partir de cette définition et des propriétés de MP et GEN on a

Theorem 3

1. Dans un modèle d'une théorie du premier ordre tous les théorèmes sont vrais. Cela provient de la validité de MP et GEN.
2. Si une théorie a un modèle alors elle est consistante. Si ce n'était pas le cas on aurait deux théorèmes w et $\neg$ w, mais comme tous les théorèmes sont vrais ce n'est pas possible.

5 Manipulations syntaxiques

5.1 Formes normales et prenex

Une ou-clause est une formule composée de disjonctions de littéraux positifs ou négatifs.

Par ex. $P(x, y) \vee Q(x, z, u) \vee \neg R(t, x)$.

Une et-clause est une formule composée de disjonctions de littéreaux positifs ou négatifs.

Une formule est sous forme normale conjonctive (FNC) si c'est la conjonction de ou-clauses, elle est en forme normale disjonctive (FND) si c'est la disjonction de et-clauses.

L'intérêt des ces formes normales réside dans le théorème suivant:

Theorem 4 Si w est une formule sans quantificateurs il existe un algorithme qui fournit une formule équivalente w' qui est en FNC. Un algorithme similaire fournit une formule équivalente en FND.

Une *clause de Horn* est une ou-clause qui ne contient qu'un seul littéral positif.

Pour normaliser des formules avec quantificateurs nous devons introduire la forme *Prenex*.

Une formule est dite en forme Prenex si tous les quantificateurs apparaissent au début.

Theorem 5 Il est toujours possible de transformer une formule en une formule équivalente sous forme Prenex.

Pour mettre une formule sous forme Prenex, on applique autant de fois que nécessaire les équivalences suivantes:

$\neg(\forall x.w_1) = \exists x.(\neg w_1)$,

$\neg(\exists x.w_1) = \forall x.(\neg w_1)$,

$(\forall x.w_1 \wedge w_2) = \forall x.(w_1 \wedge w_2)$, (si x n'apparait pas dans w_2)

$(\forall x.w_1 \vee w_2) = \forall x.(w_1 \vee w_2)$, (si x n'apparait pas dans w_2)

Exemple.

$\forall x.\forall y.(\text{Enseigne}(x, y) \Rightarrow \exists z.\text{Diplomé}(x, z))$

$= \forall x.\forall y.(\neg\ \text{Enseigne}(x, y) \vee \exists z.\text{Diplomé}(x, z))$

$= \forall x.\forall y.\exists z.(\neg\ \text{Enseigne}(x, y) \vee \text{Diplomé}(x, z))$

$= \forall x.\forall y.\exists z.(\text{Enseigne}(x, y) \Rightarrow \text{Diplomé}(x, z))$.

5.2 Skolemisation et clauses de Horn

La skolemisation d'une variable x existentiellement quantifiée dans une formule en forme Prenex consiste à remplacer x par une fonction, dite de Skolem, qui dépend des variables universellement quantifiées qui précèdent x dans la liste des quantificateurs.

Exemple. La skolemisation de z dans

$$w = \forall x.\forall y.\exists z.(\text{Enseigne}(x, y) \Rightarrow \text{Diplomé}(x, z))$$

donne la formule

$$w_s = \forall x.\forall y.(\text{Enseigne}(x, y) \Rightarrow \text{Diplomé}(x, f(x, y)))$$

La formule obtenue par skolemisation n'est pas équivalente à la formule de départ, mais elle possède des propriétés intéressantes exprimées dans le théorème ci-dessous.

Theorem 6 Soit w une formule et w_s sa skolemisation, alors

(1) tout modèle de w_s est une modèle de w

(2) si w possède un modèle alors w_s aussi

(3) w est satisfaisable si et seulement si w_s l'est (découle de (1) et (2))

Ainsi, par une succession de skolemisations on peut transformer une formule en une formule sans quantificateurs existentiels qui possède les mêmes caractéristiques du point de vue de la satisfaisabilité. Une telle formule est appelée formule universelle.

6 Univers de Herbrand

Soit w une formule universelle, et soit W le vocabulaire utilisé dans w (ensemble des symboles de constantes, fonctions et prédicats). Si l'ensemble des constantes de W est vide on ajoute à W un nouveau symbole de constante a.

L'*ensemble des constantes* de w est défini comme le sous-ensemble de W formé uniquemement des symboles de constantes de W.

Le langage formé de tous les termes que l'on peut former sur W et qui ne contiennent pas de variables est appelé *univers de Herbrand* de w.

L'ensemble A des formules atomiques sur W et qui ne contiennent pas de variables est appelé *ensemble des atomes* de w.

Si l'on considère la formule

$$w_s = \forall x.\forall y.(\text{Enseigne}(x, y) \Rightarrow \text{Diplomé}(x, f(x, y))),$$

son ensemble de constantes est {a} (car il n'y a aucune constante dans w_s), son univers de Herbrand est

$$H = \{f(a,a), f(f(a,a),a), f(a,f(a,a)), f(f(f(a,a),a),a), \ldots\}$$

(il est infini) et son ensemble d'atomes est

$$A = \{\text{Enseigne}(t_1, t_2) \mid t_1, t_2 \in H\} \cup \{\text{Diplomé}(t_1, t_2) \mid t_1, t_2 \in H\}.$$

Remarque. Si w contient un ou plusieurs symboles de fonctions son univers de Herbrand est toujours infini.

Considérons maintenant une formule $w(x_1, \ldots, x_n)$ sans quantificateurs où $x_1, \ldots, x_n$ sont libres. Si l'on remplace chaque variable par un terme de H, on obtient une formule fermée dont les atomes appartiennent à A.

Par exemple, si w = Enseigne(x, y) $\Rightarrow$ Diplomé(x, f(x, y)), la substitution (a/x, f(a)/y) produit la formule fermée (proposition) α = Enseigne(a, f(a)) $\Rightarrow$ Diplomé(a, f(a, f(a))).

On peut considérer que α est une formule de logique propositionnelle dont les variables sont 'Enseigne(a, f(a))' et 'Diplomé(a, f(a, f(a)))', qui appartiennent à A.

Si l'on fait une autre substitution on obtiendra une autre formule α' dont les variables appartiennent à A.

Le théorème de Herbrand nous dit que la formule de départ $\forall x_1. \ldots, \forall x_n w(x_1, \ldots, x_n)$ est satisfaisable pour autant qu'on puisse donner à chaque atome de A une valeur de vérité (*vrai* ou *faux*) telle que toute formule α créée comme ci-dessus soit vraie. On ramène donc le problème de chercher un modèle pour une formule de logique des prédicats au problème de chercher un modèle pour un ensemble (infini) de formules de logique des propositions.

Theorem 7 Soit $w(x_1, \ldots, x_n)$ une formule sans quantificateurs où $x_1, \ldots, x_n$ sont libres et soit la formule universelle

$$g = \forall x_1. \ldots, \forall x_n.w.$$

Soit H l'univers de Herbrand de g et A son ensemble d'atomes. Alors g est satisfaisable si et seulement si l'ensemble de formules propositionnnelles

$$O = \{w(t_1/x_1, \ldots, t_n/x_n) \mid t_1, \ldots, t_n \in H\}$$

est satisfaisable.

L'intérêt pratique de ce théorème n'est pas évident car il s'agit de trouver un modèle pour un ensemble infini de formules. Par contre, si on veut démontrer la non satisfaisabilité il suffit de considérer des sous-ensembles finis de O, comme l'indique le théorème suivant.

Theorem 8 (Herbrand) g n'est pas satisfaisable ssi il existe une proposition $\omega_1 \wedge \omega_2 \wedge \ldots \wedge \omega_n$ telle que $\{\omega_1, \omega_2, \ldots, \omega_n\}$ est un sous ensemble *fini* de O et $\omega_1 \wedge \omega_2 \wedge \ldots \wedge \omega_n$ n'est pas satisfaisable

Conséquence du théorème de Herbrand

Pour tout vocabulaire W l'ensemble des formules fermées non satisfaisable est récursivement énumérable, c'est à dire qu'il existe un programme (semi-algorithme) qui pour toute formule g

- s'arrête et répond "non" si γ n'est pas satisfaisable
- ne s'arrête pas si γ est satisfaisable

Le théorème ne donne cependant pas une méthode efficace pour choisir l'ordre d'exploration des termes possible, les méthodes de résolutions tendent à limiter cet espace de recherche. Le théorème ne fournit pas non plus un véritable algorithme qui s'arrêterait à coup sûr en répondant "oui" ou "non". En fait, Church et Turing on démontré que

il n'existe pas d'algorithme qui,
pour toute formule de logique des prédicats,
teste si celle-ci est satisfaisable ou non.

7 Principe de résolution

Le principe de résolution est une manipulation syntaxique qui, par ajout à une formule en forme normale conjonctive

$$\kappa_1 \wedge \ldots \wedge \kappa_n$$

où $\kappa_i = \lambda_{i1} \vee \ldots \vee \lambda_{ik(i)}$ (λ_{ij}littéraux)

de nouvelles clauses, permet de montrer la non satisfaisabilité de cette formule. Chaque nouvelle clause, appelée résolvant, et dérivée à partir de deux clauses déjà présentes

Résolution de base:

soit les deux clauses

$$\kappa_1 = \lambda \vee \kappa_1' \text{ ,}$$

$$\kappa_2 = \neg \lambda \vee \kappa_2'$$

le résolvant de κ_1 et κ_2 est

$$\kappa_1' \vee \kappa_2'$$

Résolution générale (avec unification)

Soit

$$\kappa_1 = \lambda \vee \kappa_1'$$

$$\kappa_2 = \neg \mu \vee \kappa_2'$$

Soit σ la plus petite substitution telle que $\lambda\sigma = \mu\sigma$. Si une telle substitution existe on dit que λ et μ sont unifiables (et que σ est l'unificateur le plus général) et on obtient le résolvant

$$\kappa_r = \sigma\kappa_1' \vee \sigma\kappa_2'$$

Exemple

$$\kappa_1 = \neg\, p(x, y) \vee q(x)$$

$$\kappa_2 = p(a, b)$$

p(x,y) et p(a, b) sont unificables par la substitution {a/x, b/y} et donnent le résolvant

$$q(a)$$

Theorem 9 (Principe de résolution de Robinson)

Soit ζ une formule sans quantificateurs en forme normale conjonctive et

$$\gamma = \forall b_1.\forall b_2. \dots \forall b_n.\zeta$$

γ n'est pas satisfaisable ssi il y a une suite de résolutions des clauses de ζ qui conduisent à la clause vide.

Exemple

on part des clauses

$$\kappa_1 = \neg\, p(x,y) \vee q(x)$$

$$\kappa_2 = p(a, b)$$

$$\kappa_3 = \neg\, q(a)$$

par résolution de κ_1 et κ_2 on obtient

$$\kappa_4 = q(a)$$

puis par résolution de κ_3 et κ_4

$$\kappa_5 = \text{<vide>}$$

Algorithme d'unification

Pour appliquer concrètement le principe de résolution il faut trouver le plus petit unificateur de deux littéraux. On utilise pour cela l' algorithme d'unification décrit ci dessous.

Soit *exp1* et *exp2* deux expressions qui sont des littéraux ou des termes et σ une substitution (vide au premier appel).

1. si *exp1* = *exp2*, retourner σ
2. si *exp1* est une variable x et *exp2* est un terme QUI NE CONTIENT PAS x, retourner $\sigma \cup \{exp2/x\}$
3. si *exp1* = $g(u_1, \dots, u_n)$ et *exp2* = $g(v_1, \dots, v_n)$ sont deux termes fonctionnels, ou deux littéraux et si $\langle u_i, v_i\rangle$ est la première paire de termes non égaux
   a) faire l'unification de u_i, v_i et σ pour obtenir σ'
   b) faire l'unification de *exp1*σ', *exp2*σ' et σ' pour obtenir σ''
   c) retourner σ''
4. sinon ECHEC

Exemple

unifier r(g(x), y, g(g(z))) et r(u, g(u), g(v))

cas 3, première paire différente: <g(x), u>,

a) faire l'unification g(x), u, et {},

cas 2. retourner la substitution {g(x)/u}

b) faire l’unification r(g(x), y, g(g(z))), r(g(x), g(g(x)), g(v)) et {g(x)/u}

cas 3., première paire différente: <y, g(g(x))>

a) faire l’unification y, g(g(x)) et {g(x)/u}

cas 2. retourner la substitution {g(x)/u, g(g(x))/y}

b) faire l’unification r(g(x), g(g(x)), g(g(z))), r(g(x), g(g(x)), g(v)) et {g(x)/u, g(g(x))/y}

cas 3., première paire différente: <g(g(z)) et g(v)>

a) faire l’unification g(g(z)), g(v), {g(x)/u, g(g(x))/y}

cas 3., ...cas 2., ... retourner {g(x)/u, g(g(x))/y, g(z)/v}

b) faire l’unification r(g(x), g(g(x)), g(g(z))), r(g(x), g(g(x)), g(g(z))) et {g(x)/u, g(g(x))/y, g(z)/v}

cas 1. retourner {g(x)/u, g(g(x))/y, g(z)/v}

c) retourner {g(x)/u, g(g(x))/y, g(z)/v}

c) retourner {g(x)/u, g(g(x))/y, g(z)/v}

c) retourner {g(x)/u, g(g(x))/y, g(z)/v}

8 Prolog

Prolog peut être considéré à la fois comme un système de démonstration automatique et comme un langage de programmation basé sur la logique des prédicats. Prolog utilise le principe de résolution pour prouver qu’une formule est une conséquence logique d’un ensemble de formules données. Si la formule à prouver contient des variables libres, Prolog cherche toutes les valeurs des variables pour lesquelles la formule est prouvable.

Exemple. Si les formules données sont :

- p(a, b)
- p(b, b)
- p(c, a)
- q(X) :- p(X, b) (syntaxe Prolog pour signifier ∀X.p(X, b) => q(X))

et si la formule à prouver est q(Y)

Prolog fournira comme réponse Y = a et Y = b

En effet, q(a) est prouvable à partir de p(a, b) et p(a, b) => q(a), idem pour q(b).

Les caractéristiques de Prolog sont

- les formules sont des clauses de Horn (= clauses avec un seul littéral positif)
- les preuves sont faites par réfutation, en appliquant le principe de résolution

  c-à-d que pour prouver que {p(a, b), p(b, b), p(c, a), q(X) :- p(X, b)} |= q(a), on prouve que {p(a, b), p(b, b), p(c, a), q(X) :- p(X, b), ¬ q(a)} est inconsistant.
- les résolutions sont appliquées en “chaînage arrière” (on remonte à partir de la clause à réfuter) et dans un ordre dépendant de l’ordre d’écriture des clauses
- on peut contrôler le processus de résolution

- même si une clause est prouvable il n’est pas certain que Prolog trouve la preuve en un temps fini.