Logique du premier ordre (3)

F. Alexandre

École Supérieure d'Informatique et Applications de Lorraine

March 3, 2009

Definition

Une clause est une disjonction de littéraux (atomes ou négation d'atomes).

Remarques

- $r(x,\ y) \vee \neg r(f(x,\ y),\ z) \vee p(y)$ est un exemple de clause. Une clause est donc une formule sans quantificateur.
- Une formule sans quantificateur peut être considérée comme une formule du calcul des propositions dans laquelle les variables propositionnelles sont les atomes, on pourra donc utiliser l'algorithme vu dans le cas propositionnel pour mettre une formule sans quantificateur sous forme d'un ensemble de clauses.
- L'ensemble des clauses correspondant à une formule est interprété comme la conjonction des clauses de l'ensemble. Toute formule du premier ordre sans quantificateur est équivalente à une conjonction de clauses.

Les étapes suivantes vont donc consister à transformer une formule avec quantificateurs en une formule sans quantificateur.

Definition

Une formule α est **sous forme prénexe** si elle est de la forme $Q_1 x_1 \ldots Q_n x_n \; \alpha'$ où $Q_i = \forall$ ou $Q_i = \exists$ et α' est une formule sans quantificateur. $Q_1 x_1 \ldots Q_n x_n$ est le **préfixe** de α. Le préfixe peut être vide, si c'est le cas la formule est prénexe.

Example

$\forall x \exists y (r(x,\ y) \Rightarrow r(y,\ x))$ est sous forme prénexe.
$\forall x \exists y \ r(x,\ y) \Rightarrow r(y,\ x)$ n'est pas sous forme prénexe.

Théorème

Toute formule est équivalente à une formule sous forme prénexe.

On considère que la formule sur laquelle s'applique les transformations est polie.

1. Elimination de $\Leftrightarrow$ et $\Rightarrow$

$$F \Leftrightarrow G \quad = \quad F \Rightarrow G \ \wedge \ G \Rightarrow F$$

$$F \Rightarrow G \ = \ \neg F \vee G$$

2. Descente des négations jusqu'aux atomes

$$\neg(\neg F) \ = \ F$$

$$\neg(F \vee G) \ = \ \neg F \wedge \neg G$$

$$\neg(F \wedge G) \ = \ \neg F \vee \neg G$$

$$\neg \forall x F[x] \ = \ \exists x \neg F[x]$$

$$\neg \exists x F[x] \ = \ \forall x \neg F[x]$$

3. Remontée des quantificateurs dans la formule jusqu'à obtenir une formule prénexe (Q est un quantificateur)

$$QxF[x] \vee G \ = \ Qx(F[x] \vee G) \text{ (x n'apparaît pas dans G)}$$

$$QxF[x] \wedge G \ = \ Qx(F[x] \wedge G) \text{ (x n'apparaît pas dans G)}$$

Example

$\forall x \forall y(\exists z(p(x,z) \wedge p(y,z)) \Rightarrow \exists u\ q(x,y,u))$
$\forall x \forall y(\neg \exists z(p(x,z) \wedge p(y,z)) \vee \exists u\ q(x,y,u))$ (élimination de $\Rightarrow$)
$\forall x \forall y(\forall z \neg(p(x,z) \wedge p(y,z)) \vee \exists u\ q(x,y,u))$ (descente des négations)
$\forall x \forall y(\forall z(\neg p(x,z) \vee \neg p(y,z)) \vee \exists u\ q(x,y,u))$ (descente des négations)
$\forall x \forall y \forall z \exists u((\neg p(x,z) \vee \neg p(y,z)) \vee q(x,y,u))$ (remontée des quantificateurs)
$\forall x \forall y \forall z \exists u(\neg p(x,z) \vee \neg p(y,z) \vee q(x,y,u))$

Introduction.
La skolémisation est une transformation qui **élimine** les quantificateurs **existentiels** d'une formule prénexe,
et qui préserve l'existence d'un modèle.
Cette transformation introduit de nouveaux symboles de fonctions qu'on appelle symboles de Skolem.
Idée intuitive : donnons quelques exemples de skolémisation.

- $\exists x\ p(x)$ est skolémisée en $p(X)$ où X est un "nouveau" symbole de fonction (ici une constante). On a
  -$p(X) \models \exists x\ p(x)$
  -$\exists x\ p(x) \not\models p(X)$.
  Mais si $\exists x\ p(x)$ a un modèle, $p(X)$ a un modèle (on préserve ainsi l'existence d'un modèle)
- $\forall x \exists y\ p(x, y)$ est skolémisée en $\forall x\ p(x, Y(x))$ où Y est un nouveau symbole de fonction d'arité 1. On a :
  -$\forall x\ p(x, Y(x)) \models \forall x \exists y\ p(x, y)$
  -$\forall x \exists y\ p(x, y) \not\models \forall x\ p(x, Y(x))$
  Si $\forall x \exists y\ p(x, y)$ a un modèle, $\forall x\ p(x, Y(x))$ a un modèle (on préserve l'existence d'un modèle)

Definition

Soit $\alpha = Q_1x_1 \ldots Q_kx_k\ \alpha'$ une formule prénexe dont l'ensemble des variables libres est $\{y_1, \ldots, y_r\}$,
on suppose que $Q_1x_1 \ldots Q_sx_s$ contient seulement des quantificateurs universels et que $Q_{s+1}x_{s+1}$ soit le quantificateur $\exists x_{s+1}$.
Soit un nouveau symbole fonctionnel φ d'arité $r + s$.
La skolémisation de la variable x_{s+1} consiste à remplacer dans α' toutes les occurrences de x_{s+1} par le terme $\varphi(y_1, \ldots, y_r, x_1, \ldots, x_s)$.
La transformée de Skolem de α est obtenue en skolémisant toutes les variables existentielles de α, on la note α^s.

Example

$F = \exists x \forall y \forall z \exists u \forall v \exists w\ p(x, y, z, u, v, w)$

$\forall y \forall z \exists u \forall v \exists w\ p(X, y, z, u, v, w)$ (skolémisation de x)

$\forall y \forall z \forall v \exists w\ p(X, y, z, U(y, z), v, w)$ (skolémisation de u)

$\forall y \forall z \forall v\ p(X, y, z, U(y, z), v, W(y, z, v))$ (skolémisation de w)

Théorème

Soit $\mathcal{A}$ un ensemble de formules prénexes et $\mathcal{A}^s$ l'ensemble des formules de Skolem de $\mathcal{A}$

1. pour chaque formule de $\mathcal{A}$, $\alpha^s \Rightarrow \alpha$ est un théorème de la logique du premier ordre
2. pour chaque formule de $\mathcal{A}$ et chaque interprétation I, il existe un prolongement de I aux nouveaux symboles de Skolem introduits dans α^s tel que dans la nouvelle interprétation I', α et α' aient la même valeur sémantique.
3. $\mathcal{A}$ est contradictoire si et seulement si $\mathcal{A}^s$ est contradictoire.

Remarques

1. les symboles de Skolem introduits par la transformation peuvent être vus comme des noms d'algorithmes permettant de calculer les valeurs explicites des variables existentielles
2. les symboles de Skolem introduits doivent être différents des autres symboles fonctionnels et différents entre eux

- algorithme[1] utilisant la méthode des attributs, contrairement aux approches classiques les trois étapes : mise sous forme prénexe, skolémisation, mise sous forme clausale, sont effectuées en parcourant une seule fois la formule.
- les formules de la logique du premier ordre sont définies à partir de la grammaire algébrique suivante : $(N,\ Ter,\ \rightarrow,\ Y)$
  - $N = \{Y,\ X,\ A,\ T\}$,
  - $Ter = \{\mathcal{F},\ \mathcal{V},\ \neg,\ \Rightarrow,\ \wedge,\ \vee,\ \exists,\ \forall,\ (,\ ),\ ,\} \cup V \cup F \cup R$
  - $Y \rightarrow X$

    $X \rightarrow (X)\ |\neg X\ |X \vee X\ |X \wedge X\ |X \Rightarrow X\ |\exists x(X)\ |\forall x(X)\ |A$

    $A \rightarrow \mathcal{F}\ |\mathcal{V}\ |r(T, \ldots, T)$

    - r est un élément de R, un symbole de relation

    $T \rightarrow x\ |a\ |f(T, \ldots, T)$

    - x est élément de V, un symbole de variable
    - a et f sont des symboles de F, des symboles de fonctions d'arités respectives 0 et strictement supérieure à 0
- les autres connecteurs logiques s'expriment en utilisant $\neg$ et $\vee$

[1] MATHÉMATIQUES DISCRÈTES. Automates, langages, logique et décidabilité. Pierre Marchand. DUNOD. Pierre.Marchand@esial.uhp-nancy.fr

- Spécification :
  - Données : α une formule de la logique du premier ordre sous forme polie, on suppose de plus que l'on a calculé l'ensemble $VL(\alpha)$ des variables libres de la formule α
  - Résultat : $C(\alpha)$ un ensemble de clauses dont la conjonction est équivalente à α
- mise sous forme d'un arbre abstrait de la formule α
- attributs considérés :
  - ν attribut qui compte le nombre de négations qui porte sur chaque nœud de l'arbre abstrait de la formule. Comme deux négations "s'annulent" (i.e. $\neg\neg p \Leftrightarrow p$) il suffit de compter le nombre de négations "modulo" 2. Cet attribut est hérité, c'est à dire qu'on propage sa valeur dans l'arbre abstrait en "descendant" dans l'arbre.
  - λ attribut qui calcule la liste des variables libres ou universellement quantifiées rencontrées le long de la branche menant de la racine de l'arbre au nœud considéré. λ est un attribut hérité.
  - σ attribut qui calcule la substitution permettant de substituer les variables existentiellement quantifiées. σ est un attribut hérité.
  - C attribut qui calcule l'ensemble de clauses. La valeur de C est propagée en remontant dans l'arbre en partant des atomes jusqu'à la racine, C est un attribut synthétisé.
  - a est un attribut qui construit les atomes. La valeur de a est propagée en remontant dans l'arbre en partant des variables jusqu'aux symboles de tête des atomes. a est un attribut synthétisé.

- Définition de l'attribut ν

Règle	$Y \rightarrow X$	$X_1 \rightarrow \neg X_2$	$X_1 \rightarrow X_2 \vee X_3$
Calcul de ν	$\nu(X) = 0$	$\nu(X_2) = 1 - \nu(X_1)$	$\nu(X_2) = \nu(X_1)$ $\nu(X_3) = \nu(X_1)$
Règle	$X_1 \rightarrow X_2 \wedge X_3$	$X_1 \rightarrow X_2 \Rightarrow X_3$	$X_1 \rightarrow (X_2)$
Calcul de ν	$\nu(X_2) = \nu(X_1)$ $\nu(X_3) = \nu(X_1)$	$\nu(X_2) = 1 - \nu(X_1)$ $\nu(X_3) = \nu(X_1)$	$\nu(X_2) = \nu(X_1)$
Règle	$X \rightarrow A$	$X_1 \rightarrow \exists x X_2$	$X_1 \rightarrow \forall x X_2$
Calcul de ν	$\nu(A) = \nu(X)$	$\nu(X_2) = \nu(X_1)$	$\nu(X_2) = \nu(X_1)$

- Les règles sont quasiment identiques au cas de la logique propositionnelle. Le calcul de l'attribut ν s'arrête au niveau des atomes (il faut savoir si on va considérer l'atome ou la négation de l'atome)

- Attributs λ (variables libres ou universellement quantifiées), σ (substitution des variables existentielles)

Règle	$Y \rightarrow X$	$X_1 \rightarrow \neg X_2$ ou $X_1 \rightarrow (X_2)$	$X_1 \rightarrow X_2\ op\ X_3$ $op \in \{\vee,\ \wedge,\ \Rightarrow\}$
Calcul de λ Calcul de σ	$\lambda(X) = VL(\alpha)$ $\sigma(X) = \emptyset$	$\lambda(X_2) = \lambda(X_1)$ $\sigma(X_2) = \sigma(X_1)$	$\lambda(X_i) = \lambda(X_1)\ i \in \{2,\ 3\}$ $\sigma(X_i) = \sigma(X_1)\ i \in \{2,\ 3\}$

Règle	$X_1 \rightarrow \exists x(X_2)$
Calcul de λ Calcul de σ	$\lambda(X_2) =$ si $\nu(X_1) = 0$ alors $\lambda(X_1)$ sinon $\lambda(X_1) \cup \{x\}$ $\sigma(X_2) =$ si $\nu(X_1) = 0$ alors $\sigma(X_1) \cup \{x \rightarrow new(\lambda(X_1))\}$ sinon $\sigma(X_1)$

Règle	$X_1 \rightarrow \forall x(X_2)$
Calcul de λ Calcul de σ	$\lambda(X_2) =$ si $\nu(X_1) = 1$ alors $\lambda(X_1)$ sinon $\lambda(X_1) \cup \{x\}$ $\sigma(X_2) =$ si $\nu(X_1) = 1$ alors $\sigma(X_1) \cup \{x \rightarrow new(\lambda(X_1))\}$ sinon $\sigma(X_1)$

Règle	$X \rightarrow A$	$A \rightarrow r(T_1, \ldots, T_k)$	$T \rightarrow f(T_1, \ldots, T_k)$
Calcul de λ Calcul de σ	$\lambda(A) = \lambda(X)$ $\sigma(A) = \sigma(X)$	$\lambda(T_i) = \lambda(A)$ $\sigma(T_i) = \sigma(A)$	$\lambda(T_i) = \lambda(T)$ $\sigma(T_i) = \sigma(T)$

- *new* est une procédure qui, à chaque appel, génère un nouveau symbole de fonction.

Définition des attributs C (ensemble de clauses) et a (construction des atomes)

Règle	$Y \rightarrow X$	$X_1 \rightarrow \neg X_2$ $X_1 \rightarrow (X_2)$ $X_1 \rightarrow \exists x X_2$ $X_1 \rightarrow \forall x X_2$	$X_1 \rightarrow X_2 \; op \; X_3$ $op \in \{\vee, \;\; \Rightarrow\}$
Calcul de C	$C(Y) = C(X)$	$C(X_1) = C(X_2)$	$C(X_1) =$ si $\nu(X_1) = 0$ alors $C(X_2) \otimes C(X_3)$ sinon $C(X_2) \cup C(X_3)$
Règle	$A \rightarrow \mathcal{V}$	$A \rightarrow \mathcal{F}$	$X_1 \rightarrow X_2 \wedge X_3$
Calcul de C	$C(A) =$ si $\nu(A) = 0$ alors $\emptyset$ sinon $\{\square\}$	$C(A) =$ si $\nu(A) = 0$ alors $\{\square\}$ sinon $\emptyset$	$C(X_1) =$ si $\nu(X_1) = 0$ alors $C(X_2) \cup C(X_3)$ sinon $C(X_2) \otimes C(X_3)$
Règle	$X \rightarrow A$	$A \rightarrow r(T_1, \ldots, T_k)$	$T_0 \rightarrow f(T_1, \ldots, T_k)$
Calcul de C Calcul de a	$C(X) = C(A)$	$C(A) =$ si $\nu(A) = 0$ alors $\{r(a(T_1), \ldots, a(T_k))\}$ sinon $\{\neg r(a(T_1), \ldots, a(T_k))\}$	 $a(T_0) = f(a(T_1), .., a(T_k))$
Règle	$T \rightarrow a$	$T \rightarrow x$	
Calcul de a	$a(T) = a$	$a(T) =$ si $x \in \sigma(T)$ alors $\sigma(T)(x)$ sinon x	

L'attribut a agit dans l'arbre abstrait au niveau des termes.

L'attribut C a le même rôle que dans le cas propositionnel (la seule différence est que les clauses sont constituées d'atomes ou de négations d'atome)

- Mettre sous forme clausale la formule $F = \neg\forall x \exists y \forall z (r(x,\ z) \Rightarrow r(x, y))$
- Mettre sous forme clausale la formule $G = \neg\{\exists y \forall x[(r(x, y) \Rightarrow r(x, x)) \wedge (r(x, x) \Rightarrow r(x, y))] \Rightarrow \forall u \exists v \forall z[(r(z, v) \Rightarrow \neg r(z, u)) \wedge (\neg r(z, u) \Rightarrow r(z, v))]\}$

Objectif : montrer que

$$\mathcal{A} \models \alpha$$

$\mathcal{A}$ un ensemble de formules closes, α une formule close (si ces formules ne sont pas closes on les rend closes en prenant leurs clôtures universelles)

- utiliser le résultat suivant : $\mathcal{A} \models \alpha$ si et seulement $\mathcal{A} \cup \{\neg\alpha\}$ est contradictoire
- utiliser l'algorithme de mise sous forme clausale et de skolémisation, pour mettre les formules de $\mathcal{A} \cup \{\neg\alpha\}$ sous forme de clauses, on obtient ainsi l'ensemble de clauses $C(\mathcal{A}) \cup C(\neg\alpha)$
- démontrer que l'ensemble de clauses $C(\mathcal{A}) \cup C(\neg\alpha)$ est contradictoire en utilisant le système formel basé sur les clauses en générant la clause $\square$.

Definition

Soit $\mathcal{L}$ un langage de la logique du premier ordre. Soit $\mathcal{C}l(\mathcal{L})$ l'ensemble des clauses construites sur $\mathcal{L}$. Le système formel $\mathcal{S} = (\mathcal{C}l(\mathcal{L}),\ \emptyset,\ \mathcal{R})$ suivant est appelé système de Robinson.
$\mathcal{R} = \{resolution,\ fact^{+},\ fact^{-}\}$

- règle de résolution
$$\frac{c_1 \vee r(s_1,\ldots,s_n) \vee c_2,\quad c_3 \vee \neg r(t_1,\ldots,t_n) \vee c_4}{\sigma(c_1 \vee c_2 \vee c_3 \vee c_4)}(resolution)$$
- règle de factorisation positive
$$\frac{c_1 \vee r(s_1,\ldots,s_n) \vee c_2 \vee r(t_1,\ldots,t_n) \vee c_3}{\sigma(c_1 \vee c_2 \vee c_3)}(fact^{+})$$
- règle de factorisation négative
$$\frac{c_1 \vee \neg r(s_1,\ldots,s_n) \vee c_2 \vee \neg r(t_1,\ldots,t_n) \vee c_3}{\sigma(c_1 \vee c_2 \vee c_3)}(fact^{-})$$

-c_1, c_2, c_3, c_4 sont des clauses
-$r(s_1,\ldots,s_n)$ et $r(t_1,\ldots,t_n)$ sont des atomes
-σ est un unificateur principal de $r(s_1,\ldots,s_n)$ et de $r(t_1,\ldots,t_n)$

Remarques

- Dans la pratique c'est la règle de résolution qui suffit la plupart du temps à générer la clause $\square$, les règles de factorisation sont très peu utilisées. **Attention** : lorsque l'on applique la règle de résolution entre deux clauses, il faut renommer les variables de façon à ce que les deux clauses n'aient pas de variables en commun.
- La règles de résolution fait intervenir deux clauses, alors que les règles de factorisation ne portent que sur une clause.
- Le système formel de Robinson n'a pas d'axiome.

Théorème de Robinson

Un ensemble C de clauses est contradictoire si et seulement s'il existe une démonstration de $\square$ avec hypothèses dans C.

On a donc les équivalences suivantes, si $\mathcal{A}$ est un ensemble de formules closes et α une formule close.

$$\begin{gathered} \mathcal{A} \models \alpha \\ \textit{ssi} \\ C(\mathcal{A}) \cup C(\neg\alpha) \textit{ est contradictoire} \\ \textit{ssi} \\ C(\mathcal{A}) \cup C(\neg\alpha) \vdash_{\mathcal{S}} \square \end{gathered}$$

Démonstration

Ce théorème établit la validité et une forme de complétude du système formel basé sur la résolution. La complétude exprime que si un ensemble de clauses est contradictoire il est possible de trouver une démonstration menant à la clause vide $\square$.

- Montrer que $\{\alpha,\ \beta\} \models \gamma$ où $\alpha = \forall x \exists y\ p(x,\ y)$, $\beta = \forall u \forall v\ (p(u,\ v) \Rightarrow q(u))$ et $\gamma = \forall z\ q(z)$
- Démontrer le syllogisme suivant : tout homme est mortel, Socrate est un homme donc Socrate est mortel.