Chapitre 1 Calcul des Prédicats

La représentation de la connaissance en IA par de simples moyens traditionnels tels que tableaux, structures, fichiers etc.... s'avère inefficace et/ou trop difficile. Le calcul des prédicats est un langage formel au moyen duquel des expressions très diverses peuvent être représentées. (Il existe d'autres formalismes de représentation). Avant de définir ce langage (du 1" ordre), nous définissons le langage de calcul de prédicat d'ordre 0 (ou logique propositionnelle).

Partie 1 : Logique propositionnelle :

La logique propositionnelle est la plus simple logique symbolique. Nous nous intéressons à des expressions dont chacune a la valeur vraie ou fausse mais pas les 2 en même temps.

Syntaxe :

Définition Une proposition est une expression déclarative qui peut avoir la valeur vraie ou fausse mais pas les 2. On représente ces propositions par des symboles P,Q,R ... etc. On les appelle des atomes ou formules atomiques.

Exemple : L'expression « Ali est ingénieur » peut être représentée par la proposition P.

Formules : A partir des propositions on peut former de nouvelles expressions appelées formules, en utilisant les connecteurs logiques standards: ¬(négation), ^ (conjonction et), v (disjonction ou), ⇒ (implication), et ⇔(l'équivalence). Ces nouvelles expressions sont appelées des formules.

Définition : Une formule en logique propositionnelle est définie comme suit :

-Tout atome est une formule.

-Si P et Q sont des formules alors $\neg P$, $P \vee Q$, $P \wedge Q$, $P \Rightarrow Q$, $P \Leftrightarrow Q$ sont des formules.

Définition : On définit un littéral comme étant un. atome ou sa négation.

Il est parfois nécessaire d'insérer des parenthèses pour définir des priorités entres ces différents connecteurs. L'ordre de priorité de ces connecteurs est donné par l'ordre décroissant suivant : $\Leftrightarrow\Rightarrow\wedge\vee\neg$. Lorsque le meme connecteur se répète la priorité est de gauche à droite.

Exemple : $P \Leftrightarrow Q \wedge R$ est équivalente à $(P \Leftrightarrow (Q \wedge R))$

$P \Rightarrow Q \wedge \neg R \vee S$ est équivalente à $(P \Rightarrow (Q \wedge ((\neg R) \vee S)))$

$P \wedge Q \wedge R \wedge S$ est équivalente à $((P \wedge Q) \wedge R) \wedge S$

Sémantique: La sémantique définit les règles de détermination de la vérité d'un énoncé dans le cadre d'un modèle particulier. Dans cette logique, le modèle détermine la valeur de vérité (Vrai ou Faux) de chaque symbole propositionnelle.

Valeur de vérité :Les valeurs de vérité (vraie ou fausse) des formules dépendent des valeurs de vérité de ses composantes (atomes). On peut déterminer les valeurs de vérité des formules en utilisant le tableau suivant :

G	**H**	**¬G**	**G∧H**	**G v H**	**G⇒H**	**G⇔H**
V	V	F	V	V	V	V
V	F	F	F	V	F	F
F	V	V	F	V	V	F
F	F	V	F	F	V	V

Définition: L'affectation des valeurs de vérité aux atomes P, Q, R... composant une formule est appelée interprétation. Pour une formule composée de n atomes on peut avoir 2^n interprétations possibles.

Définition : Une formule G est dite vraie sous une interprétation I si et seulement si la valeur de vérité de G est évaluée à Vraie sous cette interprétation I.

Définition : Si une formule est toujours Vraie ∀ l'interprétation I alors cette formule est dite valide et est appelée une tautologie.

Définition : Si une formule est toujours Fausse ∀ l'interprétation I alors cette formule est dite inconsistante (ou insatisfiable).

Propriétés :

- Une formule est valide ssi sa négation est inconsistante
- Une formule est inconsistante ssi sa négation est valide
- Une formule est invalide ssi il existe au moins une interprétation sous laquelle la formule est fausse.
- Une formule est consistante ssi il existe au moins une interprétation sous laquelle la formule est Vraie.
- Si une formule est valide alors elle est consistante mais pas vice versa.
- Si une formule est inconsistante alors elle est invalide mais pas vice versa.

Exemple : $P \wedge \neg P$ *est inconsitente (donc invalide)*
$P \vee \neg P$ *est valide (donc consistante)*

Corollaire : le nombre d'interprétation d'une formule étant fini, on peut toujours décider si une formule, est valide ou non, en examinant toutes les interprétations possibles. L'algorithme qui nous permet de déterminer la valeur de vérité d'une formule de n atomes est de l'ordre $0(2^n)$. Cet algorithme est valide (puisqu'il met en œuvre la définition de la notion de conséquence logique) et est complet puisqu'il a un nombre fini de possibilités.

Forme Normale en logique propositionnelle

Il est parfois utile de transformer une formule en une formule équivalente plus simple formée que de conjonction (ou de disjonction) de littéraux, c'est ce qu'on appelle Forme Normale Conjonctive FNC (Disjonctive FND respectivement).

Définition : 2 formules F et G sont dites équivalentes notée $F \equiv G$ ssi les valeurs de vérité de F et G sont identiques $\forall$ l'interprétation I de F et de G.

Toute formule peut être transformée sous forme normale en utilisant les règles suivantes:

1) $F \Leftrightarrow G \equiv F \Rightarrow G \wedge G \Rightarrow F$
2) $F \Rightarrow G \equiv \neg F \vee G$
3) $F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H)$ $\quad F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$
4) $\neg(\neg G) \equiv G$
5) $\neg(F \vee G) \equiv \neg F \wedge \neg G$ $\quad \neg(F \wedge G) \equiv \neg F \vee \neg G$
6) $F \vee Faux \equiv F$ $\quad F \wedge Faux \equiv Faux$ $\quad F \vee Vraie \equiv Vraie$ $\quad F \wedge Vraie \equiv F$
7) $F \vee \neg F \equiv Vraie$ $\quad F \wedge \neg F \equiv Faux$

Procédure de transformation de formule en FN:

Utiliser les règles 1 et 2, puis les règles 4 et 5 puis les règles 3

Conséquence logique: On a souvent à décider si une expression provient d'autres expressions. Ceci mène au concept de conséquence logique,

Définition: Soient des formules FI,F2,..., Fn et une formule G. G est dite conséquence logique de l'ensemble FI, F2, .. Fn (ou G provient logiquement de l'ensemble) ssi toute interprétation I dans laquelle $F1 \wedge F2 \wedge \ldots \wedge Fn$ est vraie alors G est aussi vraie. Les Fi sont des axiomes appelés les prémisses de G.

Théorème: Soient des formules FI, ... Fn et G. G est une conséquence logique de FI, F2, ... Fn ssi la formule $((F1 \wedge F2 \wedge \ldots \wedge Fn) \Rightarrow G)$ est valide.

Preuve Exercice

Théorème : G est une conséquence logique de F1, F2, … Fn ssi $(F1 \wedge F2 \ldots \wedge Fn \wedge \neg G)$ est inconsistante.

Preuve: exercice

Si G est une conséquence logique de l'ensemble FI, F2,... Fn alors la formule $((F1 \wedge \ldots . \wedge Fn) \Rightarrow G)$ est dite un théorème et G est appelé sa conclusion.

Pour montrer si une formule est conséquence logique d'un ensemble de formule on peut utiliser la définition ou l'un des 2 théorèmes précédents.

Exemple: Montrer que ¬P est conséquence logique de { P ⇒Q , ¬Q}

Partie2 : Logique d'ordre 1- calcul des prédicats d'ordre 1 :

Nous allons définir un langage plus expressif que la logique propositionnelle. Il est construit sur des objets et des relations. Les éléments de base de la logique du 1er ordre sont les symboles qui représentent les objets, les fonctions et les relations.

Syntaxe : Les composantes élémentaires dans ce langage sont les prédicats, les variables, les fonctions et les constantes séparés par des « , » et entourés par des « (,) » ainsi que les connecteurs logiques et les quantificateurs.

➔ *formules bien formées* (fbf)

Définitions:

Un Terme: est une expression logique qui renvoie à un objet .

- Une constante est un terme utilisé pour exprimer des objets constants,
- Une variable est un terme, utilisée pour représenter des entités variables,
- Si f est un symbole de fonction (foncteur) et t_1, ..., t_n sont des termes alors $f(t_1,...,t_n)$ est un terme. f est une fonction appliquée à un ensemble de termes.

Une Formule Atomique

Une formule atomique est représentée par $p(t_1, ..., t_n)$ où p est un symbole de prédicat et les t_i des termes. Un prédicat est utilisé en général pour représenter une relation dans un certain contexte. N est appelée l'arité du prédicat p

Exemple1 ***:*** La phrase *Ali est allé à* l'université peut etre représentée par : est*_allé_a(Ali, université)*

Remarque : On remarque qu'une formule atomique et le terme fonction ont la même syntaxe, mais ils auront des sémantiques différentes.

Un littéral: Un littéral est défini comme une formule atomique ou sa négation.

Formule Bien Formée (fbf):

- Toute formule atomique est une fbf,
- Toute conjonction de fbf est une fbf,
- Toute disjonction de fbf est une fbf,
- La négation d'une fbf est une fbf,
- Si p et q sont 2 fbfs alors p ⇒ q est une fbf,
- Si p et q sont 2 fbfs alors p ⇔ q est une fbf.

Exemple : Soit la phrase, Ali a un chien et Mohamed aime le chat. Cette phrase peut être représentée par : a(Ali, chient) ∧ aime(Mohamed,chat).

Les quantificateurs

Une formule atomique P(x) où x est une variable, peut avoir la valeur Vraie pour toute valeur prise par x ou pour au moins une valeur de x d'un certain domaine. Ces notions sont exprimées en utilisant les quantificateur $\forall$ et $\exists$. Une formule contenant le quantificateur universel ($\forall$ x) devant une formule P(x) a la valeur Vraie pour toutes les affectations de x aux valeurs de ce domaine. Une formule contenant le quantificateur existentiel ($\exists$x) devant une formule P(x) a la valeur Vraie pour une affectation de x à une valeur du domaine.

Exemple : La phrase « Il y a un enseignant qui assure le cours » peut être représentée par la formule: ($\exists$ x) [enseignant(x) $\wedge$ assure(x, cours)].

Définition d'une Variable liée et Variable libre

On dit que les formules des exemples précédents sont quantifiées. La variable x est la variable quantifiée. Cette variable est dite *liée.* La portion de la formule à laquelle le quantificateur s'applique est appelée portée du quantificateur. Toute variable qui n'est pas liée est *libre.*

Toute expression obtenue en quantifiant une variable dans une fbf est également une fbf. On ne s'intéresse qu'aux expressions dont toutes les variables sont liées.

Sémantique:

Définir la sémantique $\equiv$ définir la notion d'interprétation. Pour donner un sens a une formule atomique:

- On définit un domaine,
- A chaque symbole de prédicat on lui attribue une relation dans ce domaine,
- A chaque symbole de foncteur (fonction) on lui attribue une fonction dans ce domaine,
- A chaque symbole de constante on lui attribue une constante dans ce domaine,

Propriétés des fbf

Soit une interprétation I. Les valeurs de vérité des expressions ne contenant pas de variables peuvent être calculées en utilisant à la table de la partie 1 ci-dessus.

Soient X1 et X2 2 fbfs quelconques, on peut établir les relations suivantes :

$\neg(\neg X1) \equiv X1$ $\qquad$ $X1 \Rightarrow X2 \equiv \neg X1 \vee X2$

Loi de De Morgan

$\neg(X1 \wedge X2) \equiv \neg X1 \vee \neg X2$ $\qquad$ $\neg(X1 \vee X2) \equiv \neg X1 \wedge \neg X2$

Lois distributives

$X1 \wedge (X2 \vee X3) \equiv (X1 \wedge X2) \vee (X1 \wedge X3)$ $\qquad$ $X1 \vee (X2 \wedge X3) \equiv (X1 \vee X2) \wedge (X1 \vee X3)$

Lois commutatives

$X1 \wedge X2 \equiv X2 \wedge X1$ $\qquad$ $X1 \vee X2 \equiv X2 \vee X1$

$(X1 \wedge X2) \wedge X3 \equiv X1 \wedge (X2 \wedge X3)$ $\qquad$ $(X1 \vee X2) \vee X3 \equiv X1 \vee (X2 \vee X3)$

Loi de Contraposition

$X1 \Rightarrow X2 \equiv \neg X2 \Rightarrow \neg X1$

Autres Équivalences:

$\neg [(\exists x)P(x)] \equiv (\forall x) \neg P(x)$

$\neg [(\forall x) P(x)] \equiv (\exists x) \neg P(x)$

$(\forall x)[P(x)\wedge Q(x)] \equiv (\forall x)P(x) \wedge (\forall y) Q(y)$

$(\exists x)[P(x)\vee Q(x)]\equiv(\exists x)P(x)\vee(\exists y)Q(y)$

$(\forall x)P(x) \equiv (\forall y)P(y)$

$(\exists x)P(x) \equiv (\exists y)P(y)$

Définition :

Si une fbf a la valeur de vérité Vraie (respectivement Fausse) pour toutes les interprétations possibles, alors elle est dite Valide (respectivement inconsistante). Les fbf (sans variables) valides sont appelées tautologies.

On avait souligné que lorsqu'une fbf peut contenir des quantificateurs, il est parfois impossible de pouvoir déterminer la valeur de vérité de cette fbf (sur les domaines infinis par exemple). Il est démontré qu'il est impossible de trouver une méthode générale qui nous permet de décider de la validité de fbf contenant des quantificateurs. Cependant il est possible de déterminer la valeur de vérité de certaines fbf quantifiées. On parle de sous-classes décidables. *Le calcul des prédicats est semi-decidable.*

Si une même interprétation fait que chaque fbf dans un ensemble de fbf a la valeur Vraie alors nous disons que cette interprétation satisfait l'ensemble des fbf.

Définition d'une conséquence logique : Une fbf X est une conséquence logique d'un ensemble S de fbf si toute interprétation satisfaisant S satisfait X.

Règles d'inférence : Dans ce langage des prédicats, il existe des règles d'inférence qui peuvent être appliquées à certaines fbf pour produire de nouvelle fbf.

Exemples de règle d'inférence:

- Le Modus Ponens: A partir de W1 et W1 $\Rightarrow$ W2 on infère W2
- La Spécialisation universelle: A partir de $(\forall x)W(x)$ on infère W(A) où A est un symbole de constante.
- Combinaison des 2 règles précédentes: A partir de $(\forall x)[W1(x) \Rightarrow W2(x)]$ et W1(A) on infère W2(A)

Les fbf inférées sont appelées des théorèmes. La séquence d'application des règles d'inférence utilisées dans la dérivation constitue la démonstration de ce théorème.

Système de Règles d'inférence sain : Un système de règles d'inférence est dit sain si tout théorème dérivable de tout ensemble de fbf est conséquence logique de cet ensemble de fbf. On peut démontrer que le Modus Ponens est sain.

Système de Règles d'inférence complet : Un système de règles est complet si toutes les fbf qui découlent logiquement de tout ensemble de fbf sont aussi des théorèmes dérivables de cet ensemble.

Unification : Dans la règle d'inférence combinée précédente il a été nécessaire de trouver la substitution « x prend la valeur A » pour rendre W1(x) et W1(A) identique. Ce processus est appelé *Unification.* Il existe des algorithmes pour effectuer cette unification. Avant de détailler ce processus nous décrivons ce qu'est la substitution.

Définition de la substitution: Une substitution σ est une application

σ: terme ----> terme

qui est l'identité sauf en un certain nombre de point de variable. Elle est notée par l'ensemble {<X_i/t_i>}. X_i est une variable, t_i est un terme. (on la note des fois par <X_i,t_i>.

L'instanciation t' d'un terme t est définie comme étant l'application d'une substitution σ à t. On la note par t' = σ(t). Elle consiste à remplacer x_i par t_i de la substitution dans t.

Exemple : Soit la fbf P(x, f(y), B) où x, y sont des variables, B est une constante et f une fonction. On peut obtenir les instances suivantes:

P(z, f(w), B) par la substitution {x/z, y/w} (Renommage)
P(x, f(A), B) par la substitution {y/A}
P(C, f(A), B) par la substitution {x/C, y/A} (appelée Instance Close - pas de variable)

Remarque : Dans une substitution {X_i/t_i} toutes les variables sont distinctes. On suppose aussi que la variable X_i n'apparaît pas dans t_i.

<u>Composition de substitution :</u> La composition de substitution est définie comme suit :
σ1.σ2(p) =σ2(σ1(p))

Définition 2 termes t1 et t2 sont unifiables s'il existe une substitution σ telle que σ(t1)=σ(t2). Il existe des algorithmes d'unifications, dont voici un algorithme.

Procédure récursive unifier(E1,E2);
1) **Si** E1 ou E2 est un atome (prédicat, fonction, cte négation ou variable)
  alors Échanger E1 et E2 de sorte que E1 soit un atome et faire
2) **début**
3) **Si** E1 et E2 sont identiques **alors** renvoyer Nil
4) **Si** E1 est une variable
5) **alors début**
6) **Si** E1 apparaît dans E2 **alors** renvoyer Échec (Occur-check)
7) Renvoyer {E1/E2}
8) **Fin**
9) **Si** E2 est une variable **alors** renvoyer {E2/E1}
10) Renvoyer Échec
11) **Fin**
12) F1 := le premier élément de E1, T1 := le reste de E1
13) F2 := le premier élément de E2, T2 := le reste de E2;
14) Z1 := unifier(F1,F2)

15) Si Z1=Echec alors renvoyer Échec
16) G1 := Z1(T1)
17) G2 := Z1(T2)
18) Z2 := unifier(G1,G2)
19) Si Z2=Echec alors renvoyer Echec

Cet algorithme produit l'upg (l'unificateur le plus général) de 2 expressions ou echec lorsqu'elles ne sont pas unifiables.

Résolution : La résolution est une règle d'inférence importante qui peut être appliquée à une certaine classe de fbf appelées *clauses.*

Définition de Clause : Une clause est une fbf formée d'une disjonction de littéraux.

Le processus de résolution est appliqué à 2 clauses parentes pour produire une clause dérivée. Donc ce principe de résolution ne peut être appliqué qu'aux clauses. Heureusement toute fbf peut être transformée en un ensemble de clauses. Cette transformation est faite à l'aide du processus suivant.

Transformation d'une fbf en clauses.

Cette transformation comprend les étapes suivantes:

1) Eliminer les implications à l'aide de la règle suivante: $X1 \Rightarrow X2 \;\equiv\; \neg X1 \vee X2$

2) Réduire les portées des négations jusqu 'aux littéraux avec les lois de De Morgan.

3) Standardiser les variables: Dans la protée de n'importe quel quantificateur, une variable liée par ce quantificateur peut être remplacée par toute autre variable n'apparaissant pas dans la protée de ce quantificateur. (renommer les variables).

Exemple: $(\forall x)P(x) \Rightarrow (\exists x)Q(x)$ *est équivalente à* $(\forall x)P(x) \Rightarrow (\exists y)\,Q(y)$

4) Eliminer les quantificateurs existentiels par le processus suivant: Remplacer une variable existentielle par une fonction de Skolem (un nouveau nom de fonction) dont les arguments sont les variables liées à des quantificateurs universels dont la portée inclut la portée du quantificateur existentiel à éliminer. S'il n'existe pas de quantificateur universels alors la fonction de Skolem est une constante de Skolem.

Exemples :

$(\forall y)(\exists x)\,P(x,y)$ *devient* $(\forall y)P(g(y),y)$ *où g est une fonction de Skolem.*

$(\forall x)(\forall y)(\exists z)\,P(x,y,z)$ *devient* $(\forall x)P(\forall y)\,P(x,y,g(x,y))$ *où g est une fonction de Skolem*

$(\exists x)P(x)$ *devient* $P(A)$ *où A est une constante de Skolem*

5) Mettre l'expression sous forme normale prenexe. Une fbf sous forme normale prenexe est de la forme: *Prefixe* *Matrice*

Les quantificateurs $\forall$ *Formule sans les quantificateurs*

6) Mettre la matrice sous forme normale conjonctive. On utilise la règle de distributivité suivantt: $X1 \vee (X2 \wedge X3) \;\equiv\; (X1 \vee X2) \wedge (X1 \vee X3)$

7) Eliminer les quantificateurs universels (effacer la partie préfixe)

8) Eliminer les symboles $\wedge$ *en remplaçant* $X1 \wedge X2 \wedge X3 \ldots \wedge Xn$ *par l'ensemble de clauses {X1, X2, X3, ..., Xn} Chaque Xi est formée de disjonction de littéraux.*

9) Renommer les variables des clauses X_i

Exemple: Appliquer ce processus à la fbf suivante:
$(\forall x)\{P(x) \Rightarrow \{(\forall y)[P(y) \Rightarrow P(f(x,y))] \wedge \neg (\forall y)[Q(x,y) \Rightarrow P(y)]\}\}$

Résolution dans le cas général : Pour pouvoir appliquer la résolution à 2 clauses (appelées clquses parentes), il faut trouver une substitution qui puisse être appliquée aux 2 clauses de telles sorte qu'elles contiennent des littéraux complémentaires. Pour faciliter la compréhension, représentons une clause (disjonction de littéraux) sous forme d'ensemble de littéraux. Soient C1 et C2 les 2 ensembles représentant les 2 clauses parentes. Soient l1 et l2 les 2 littéraux appartenant respectivement aux clauses C1 et C2 de telles sorte qu'elle existe une substitution σ pour que $\sigma(l1)$ et $\sigma(l2)$ soient complémentaires. Nous disons alors que les 2 clauses parentes C1 et C2 se résolvent en la résolvante $\sigma(\{C1\text{-}l1\} \vee \sigma(\{C2\text{-}l2\})$. (On suppose qu'on a renommé les variables des 2 clauses parentes). (C1-l1 signifie : enlever l1 de C1)

Remarque: Si 2 clauses se résolvent, elles peuvent avoir un nombre fini de résolvante.

Exemple: Soient les 2 clauses parentes: $P(x,f(A)) \vee P(x,f(y)) \vee Q(y)$ et $\neg P(z,f(A)) \vee \neg Q(z)$ elles peuvent produire 4 résolvantes (Les littéraux complémentaires via une substitution sont soulignés):

1) $\underline{P(x, f(A))} \vee P(x, f(y)) \vee Q(y)$ et $\neg\underline{P(z, f(A))} \vee \neg Q(z)$

x/z

$P(z, f(y)) \vee Q(y) \vee \neg Q(Z)$

2) $P(x, f(A)) \vee \underline{P(x, f(y))} \vee Q(y)$ et $\neg\underline{P(z, f(A))} \vee \neg Q(z)$

x/z , y/A

$P(z, f(A)) \vee Q(A) \vee \neg Q(z)$

3) $\underline{P(x, f(A))} \vee \underline{P(x, f(y))} \vee Q(y)$ et $\neg\underline{P(z, f(A))} \vee \neg Q(z)$

y/A x/z

$Q(A) \vee \neg Q(z)$

Dans ce cas les 2 littéraux ont été réduits en un seul littéral avec la substitution.

4) $P(x, f(A)) \vee P(x, f(y)) \vee \underline{Q(y)}$ et $\neg P(z, f(A)) \vee \neg\underline{Q(z)}$

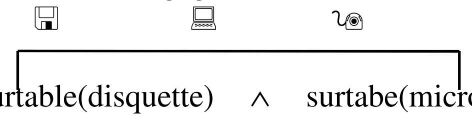
y/z

$P(x, f(A)) \vee P(x, f(z)) \vee \neg P(z, f(A))$

Propriétés: Il est démontré que la résolution est une règle d'inférence **saine**

Exemple d'utilisation du calcul des prédicats

Plusieurs états, situation ou buts en Intelligence Artificielle peuvent être représentés en utilisant ce langage. La situation suivante peut être décrite par:



surtable(disquette) ∧ surtabe(micro) ∧ surtable(souris) ∧ milieu(micro) ∧ a_gauche(disquette) ∧ a_droite(souris).