CHAPITRE 2 : CALCUL DU PREMIER ORDRE (DES PREDICATS)

Dans le chapitre précédent, nous nous sommes intéressés aux propositions comme éléments de base de notre langage. Nous allons, dans ce chapitre, enrichir notre analyse de ces propositions en introduisant une nouvelle structure, celle de prédicat.

Limites du Calcul Propositionnel

Le calcul propositionnel reste très limité, et ne permet essentiellement que d'exprimer des opérations booléennes sur des propositions, et il n'arrive pas à prouver certains raisonnements très intuitifs. Prenons les deux exemples suivants :

Exemple 1 :

Soit le raisonnement suivant :

- Hypothèses (prémisses) : Tout homme est mortel.
  Ahmed est un homme.
- Conclusion : Donc, Ahmed est mortel.

En calcul propositionnel le raisonnement précédant peut être formalisé comme suit :

soit : p : « Tout homme est mortel »
q : « Ahmed est un homme »
r : « Ahmed est mortel »

Le problème s'écrit alors $(p\wedge q)\longrightarrow r$. Cette traduction est correcte, mais elle n'est pas fidèle, car elle n'a pas pris en compte la signification du mot « tout ».

Une traduction plus correcte est possible en logique des prédicat en utilisant ce qu'on appelle des quantificateurs Les quantificateurs $\forall,\exists$ (qui ne sont pas présents en logique propositionnelle).

Exemple 2 :

Soit la proposition « x est premier ».
La valeur de vérité de cette proposition dépend de la valeur de x (qui est l'objet de la proposition). En logique propositionnelle, ce n'est pas possible de formuler une telle proposition, car la valeur de vérité d'une proposition est bien claire, elle est soit vraie, soit fausse. Cependant, en logique propositionnelle, il est possible de formuler les propositions dont la valeur de vérité dépend de l'objet.

Exemple 3
Soit la proposition : « Ahmed habite à Mila »
On ne peut pas considérer séparément :
Ahmed,
Mila,
et la relation « habiter à »

La logique des propositions traite les propositions comme un tout. Elle ne peut pas parler d'objets, de propriétés d'objets et de mettre en relation des objets.
Par contre, en calcul des prédicats, la proposition est divisée en son sujet et son prédicat. Donc :

« Ahmed habite à Mila », « Ali habite à Jijel », « Sara habite à Constantine » sont des instances d'un même moule : « *x* habite à y »
x et *y* sont des variables et habite à est le prédicat.

Le calcul des prédicats, reste le formalisme le plus courant pour exprimer des propriétés mathématiques. C'est aussi un formalisme très utilisé en informatique pour décrire les objets.

Comme pour la logique propositionnelle, on commence d'abord par la syntaxe du *Calcul du Premier Ordre* (**CPO**), et puis la sémantique.

I/ Syntaxe du CPO

1. C'est quoi un prédicat ?

En littérature, un prédicat est ce qui est affirmé d'un sujet. Par exemple, dans « Ahmed est intelligent », l'intelligence est le prédicat.

En logique mathématique, un prédicat est un moule à propositions. C'est un énoncé correct dont la valeur de vérité dépend de l'objet.

Remarque : On peut avoir :

- Un prédicat à une variable P(x) (dans ce cas, on parle d'une propriété de l'objet x)
- Un prédicat à deux variables, par exemple $x > y$. Dans ce cas, c'est une relation entre l'objet x et l'objet y
- ou encore un prédicat à n variables, et on parle alors de prédicat n-aire).

2. Alphabet du CPO

L'alphabet du calcul des prédicats est formé de :

- Des connecteurs logiques $\{\neg,\wedge,\vee,\longrightarrow,\longleftrightarrow\}$
- Des quantificateurs $\{\forall,\exists\}$
- D'un ensemble de variables $V=\{x_0, x_1, x_2, \ldots\ldots\}$
- D'un ensemble, éventuellement vide, de constantes $C=\{c_0, c_1, c_2, \ldots\}$
- D'un ensemble, éventuellement vide, de fonctions : $F_0, F_1, \ldots, F_n$

  On dit F_n est une fonctions d'arité n (nombre d'arguments=n)
- D'un ensemble de relations (prédicats) : $R_0, R_1, \ldots, R_n$
- Des deux parenthèses {(,)}.

Donc :

$$A_{CPO} = V\cup\{\neg,\wedge,\vee,\longrightarrow,\longleftrightarrow\}\cup\{\forall,\exists\}\cup\{(,)\}\cup C\cup F\cup R$$

Où :

- $V\cup\{\neg,\wedge,\vee,\longrightarrow,\longleftrightarrow\}\cup\{\forall,\exists\}\cup\{(,)\}$ est partie logique du langage (commune à tous les langage)
- $\Sigma=C\cup F\cup R$: signature du langage (partie non logique): partie qui caractérise un langage (elle est différente d'un langage à un autre)

Exemples de langage :

L1 : $\Sigma=(\{0,1\},\{+,\bullet\},\{>\})$

C$=\{0,1\}$

$F=\{+,\bullet\}=F_2$ sont deux fonctions d'arité 2

$R=\{<,=\}=R_2$ sont deux relations d'arité 2

L2 : $\Sigma=(\{0,1\},\{s,+\},\{\text{impair}, \text{premier},=,<\})$

L2 possède les constantes 0 et 1, le symbole de fonction + d'arité 2, le symbole de fonction s d'arité 1, les symboles de relations impair et premier d'arité 1, les symboles de relations = et < d'arité 2

3. Les expressions du langage

On va définir par étape d'abord :

- Les termes qui visent à représenter des objets
- Puis les formules atomiques qui visent à représenter des relations entre objets
- Et après, les formules.

a- Terme

Soit Σ =(C,F,R) une signature

l'ensemble **T** des termes sur la signature Σ est le langage sur l'alphabet $A_{CPO}(\Sigma)$ définit inductivement par :

1. Toute constante est un terme
2. Toute variable est un terme
3. Si f est un symbole de fonction à n arguments et $t_1, t_2, \ldots, t_n$ sont des termes, alors $f(t_1, t_2, \ldots, t_n)$ est un terme.

NB : un **terme clos** est un terme sans variables

b- Formule atomique (atome)

Soit Σ =(C,F,R) une signature.

Si $t_1, t_2, \ldots, t_n$ sont des termes et P un prédicat à n variables, alors $P(t_1, t_2, \ldots, t_n)$ est une formule atomique.

c- Formule

Soit Σ =(C,F,R) une signature. L'ensemble des formules $\mathcal{F}$ sur la signature Σ est le langage sur l'alphabet $A_{CPO}(\Sigma)$ défini inductivement par :

1. Toute formule atomique est une formule
2. Si F est une formule alors ¬F est une formule
3. Si F et G sont des formules alors (F∧G), (F∨G), (F⟶G) et (F⟷G) sont des formules
4. Si F est une formule et x∈V est une variable alors ∀xF et ∃xF sont des formules

Exemples :

1) $\forall x((\text{Premier}(x) \wedge x > x+11) \longrightarrow \text{impair}(x))$ est une formule sur la signature de L2

2) Σ=({f, g, t, s},{P,Q,R}

f, g ,t ,s sont des fonctions d'arité 1,2, 3,2

P,Q,R sont des relations d'arité 1,2,2

$\forall x(R(x, y)\longrightarrow Q(x, f, y))$, $\neg\exists x(R(x, y)\vee Q(x, g, y, x))$, $\forall x(P(x)\wedge\exists y(t(y, y, x)\longrightarrow s(x, y)))$ sont des formules

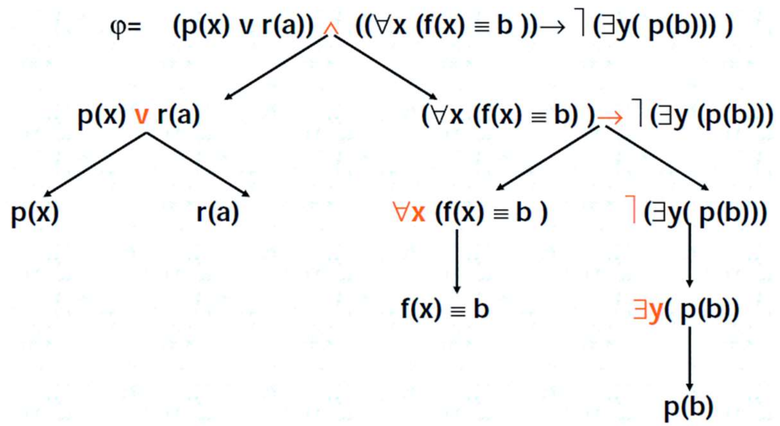
Remarques :

d - Arbre de décomposition

Une formule a aussi son *arbre de décomposition* dont les terminaisons sont des formules atomiques :

- *Procédure de décomposition*
  - On découpe au niveau du connecteur ou quantificateur principal de la formule.
  - On enlève les parenthèses les plus externes de chacun des arguments du connecteurs ou quantificateur principal.
  - On répète l'opération de décomposition pour chacun des arguments jusqu'à arriver au formules atomiques.

Exemple :



e- Les sous formules d'une formule

Les sous formules de F se définissent de la façon suivante :

-si F est une formule atomique alors SF(F)={F}
-si F=¬G alors SF(F)={F}∪SF(G)
-si F=(G*H) *∈{∧,∨,⟶,⟷} alors SF(F)={F}∪SF(G) ∪SF(H)
-si F=∀xG ou F=∃xG alors SF(F)={F}∪SF(G)

Exemples :

SF[∀x(R(x,y)⟶Q(x,f,y))]={ ∀x(R(x,y)⟶Q(x,f,y)), (R(x,y)⟶Q(x,f,y)), R(x,y), Q(x,f,y))}

4. Priorité des connecteurs

Les connecteurs et les quantificateurs sont appliqués dans l'ordre suivant :

¬, ∀, ∃, ∧,∨, ⟶,⟷,

5. Champ (portée) d'un quantificateur

Le Champ d'un quantificateur est la partie de la formule couverte par un quantificateur, autrement dit, c'est la partie sur laquelle opère ce quantificateur.

Exemple :

$$\forall x(\exists y\ (p(x,y) \to p(y,x)) \wedge x = y)$$

Champ de ∃x

Champ de ∀x

6. variables libres et liées

Une occurrence d'une variable x est liée ssi elle est dans le champ d'un quantificateur sur x (∀x, ∃x) sinon elle est libre.

Les occurrences libres de x sont définies de la façon suivante :

- Si F est une formule atomique, toutes les occurrences de x sont libres.
- Si F=¬G, les occurrences libres de x dans F sont celles de x dans G.
- Si F=(G*H), les occurrences libres de x dans F sont celles de x dans G **et** celles de x dans H
- Si F=∀yG ou F=∃yG,x≠y, les occurrences libres de x dans F sont celles de x dans G
- Si F=∀xG ou F=∃xG, // x est quantifié dans F//aucune occurrence de x dans F n'est libre, **elles sont liées**

- Une variable dans F est dite **libre** ssi elle a au moins une occurrence libre.
- Une variable est **liée** ssi toutes ses occurrences sont liées.
- Une formule F est dite **close** (ou **fermée**), si et seulement si, F ne possède pas de variables libres (toutes ses variables sont liées). Une formule est dite **ouverte** s'il a au moins une variable libre.

Exemple : Soit la formule suivante :

$$\forall y P(x,y) \wedge Q(y) \to \exists x Q(x) \to R(z)$$

Identifier les variables liées et les variables libres

$$\forall y P(x,y) \wedge Q(y) \to \exists x Q(x) \to R(z)$$

Libre Liée Liée Liée Libre

Exercice :

Pour chacune des formules, déterminer la portée de chaque quantificateur, et mettre en gras les occurrences libres :

$a)\ \exists x Axy \wedge Bx$
$b)\ \exists x \exists y\ Axy\ \rightarrow Bx$
$c)\ \exists x(\exists y Axy \rightarrow Bx)$
$d)\ \neg \exists x(Axy \vee By)x$

Corrigé :

$$\boxed{\exists x Axy} \wedge Bx$$

$$\boxed{\exists x \boxed{\exists y\ Axy}} \rightarrow Bx$$

$$\boxed{\exists x(\boxed{\exists y Axy} \ \rightarrow Bx)}$$

$$\neg \boxed{\exists x(Axy \vee By)x}$$

Remarque : On a :

- Dans (b), toutes les occurrences de « y » sont liées donc « y » est une variable liée. Il y a une occurrence libre pour « x » donc « x » est une variable libre.
- Dans (c) toutes les variables sont liées, donc (c) est une formule close.

7. Traduction en calcul des prédicats

Prenons des exemples de traduction du langage naturel en CPO :

1) Tout est relatif

Langage	Prédicat unaire **r** r(x): « x est relatif »
Formule	$\forall x \mathbf{r}(x)$

2) Rien n'est relatif
   $\forall$x ¬r (x)

3) Une porte est ouverte ou fermé

Langage	3 Prédicats unaires porte (x) : « x est une porte » ouvert(x) : « x est ouvert » fermé(x)) : « x est fermé »
Formule	$\forall x($ porte $(x) \rightarrow$ (ouvert $(x) \vee$ fermé $(x)))$

4) Tous les chemins mènent à Rome

Langage	2 Prédicats unaires chemin(x): « x est un chemin » mène-à-rome(x): « x mène à Rome »
Formule	$\forall$x(chemin (x) → mème-à-Rome (x))

Noter que la traduction n'est pas unique. On peut par exemple traduire cet énoncé comme suit :

Langage	1 Prédicat unaires ,1 prédicat binaire, 1 constante chemin(x): « x est un chemin » mène(x,y): « x mène à y » Constante: Rome
Formule	$\forall x(\text{chemin}(x) \rightarrow \text{ mene } (x, \text{Rome }))$

5) Pour tout entier il existe un entier plus grand

Langage	1 Prédicat unaires ,1 prédicat binaire e(x): « x est un entier » x > y : « x plus grand que y »
Formule	$\forall x(e(x) \rightarrow \exists y(e(y) \wedge (y > x)))$

6) Il existe un entier pair
   $\exists x\ (e(x) \wedge \text{pair}\ (x))$

7) Il existe un plus petit entier
   $\exists x\ (e\ (x) \wedge\ \forall y(\ e(y) \longrightarrow x < y))$

De la même façon :

Tous ce qui brille n'est pas or
$\exists x\ (\text{brille}\ (x) \wedge \neg\ \text{or}\ (x))$, $\neg\forall x\ (\text{brille}\ (x) \longrightarrow \text{or}\ (x))$

La traduction de logique du premier ordre en langage naturel se fait inversement :

Exemple :

$\exists x(\text{ restaurant } (x) \wedge \text{servir } (x, \text{sushi}) \wedge \text{étrePrèsDe } (x, \text{laGare }))$

Langage	2 Constantes: Sushi, laGare 1 Prédicat unaire restaurant(x): « x est un restaurant » 2 prédicats binaires Servir(x,y): « x sert y » êtrePrésDe(x,y): « x est prés de y »
Enoncé	« il existe au moins un restaurant qui sert du sushi et qui est prés de la gare »