




Programmation logique (Prolog)

I - INTRODUCTION

Le langage PROLOG est basé sur le calcul des prédicats du premier ordre (avec quelques extensions et restrictions). C'est une implémentation du *Principe de Résolution* (Robinson 1965) avec des stratégies particulières et des restrictions¹.

PROLOG a été conçu par A.Colmerauer à Marseille. Depuis, de nombreux *interpréteurs* PROLOG ont été écrits, avec des syntaxes différentes.

Dans ce qui suit, on utilisera la syntaxe et les conventions les plus couramment utilisées maintenant. Ce sont celles utilisées SWI-Prolog (utilisé dans ce module) ainsi que par, entre autres, C-Prolog, Sixtus-Prolog.




PROLOG peut être vu comme un langage d'expression de connaissances. Il peut être vu aussi comme un langage de programmation adapté au calcul symbolique et à la manipulation d'objets, de prédicats et de relations.


Il n'y a pas d'instruction, pas d'affectation, pas de boucles explicites, mais uniquement des *clauses* exprimant des faits, des règles et des questions.

Exemples de faits :

```
pere(alain, jeanne).  
pere(alain, michel).  
pere(michel, robert).  
mere(sylvie, robert).  
mere(sylvie, luc).  
pere(georges, sylvie).
```



<i>Exemples de questions :</i>	<code>:- pere(alain, jeanne)</code>	Réponses :	oui	
	<code>:- pere(alain, robert)</code>		non	
	<code>:- pere(alain, X)</code>		oui	X = michel X = jeanne
	<code>:- mere(X, robert)</code>		oui	X = sylvie
	<code>:- mere(X, michel)</code>		non	
	<code>:- pere(michel, X), mere(sylvie, X)</code>		oui	X = robert
	<code>:- pere(alain, X), pere(X, robert)</code>		oui	X = michel
	<code>:- grand_pere(alain, robert)</code>		non	




Exemple de règle, permettant de définir² la relation grand-père :

X est grand-père de Y si X est le père de quelqu'un qui est le père ou la mère de Y, c'est-à-dire s'il existe Z tel que X est le père de Z et Z est le père de Y, soit, en logique (calcul des prédicats du premier ordre) :

$$\exists Z \left(\text{père}(X,Z) \wedge (\text{père}(Z,Y) \vee \text{mère}(Z,Y)) \right) \Rightarrow \text{grand-père}(X,Y)$$

que l'on peut aussi écrire :

$$\begin{aligned} \forall Z \big(& (\text{père}(X,Z) \wedge \text{père}(Z,Y) \Rightarrow \text{grand-père}(X,Y)) \\ & \wedge (\text{père}(X,Z) \wedge \text{mère}(Z,Y) \Rightarrow \text{grand-père}(X,Y)) \big) \end{aligned}$$




Cette dernière formulation est celle qui sera utilisée en PROLOG, avec une quantification universelle implicite de toutes les variables :

```
grand_pere(X,Y) :- pere(X,Z) , pere(Z,Y) .  
grand_pere(X,Y) :- pere(X,Z) , mere(Z,Y) .
```

On peut alors interroger sur la relation grand_pere :

```
:- grand_pere(alain, robert) oui  
:- grand_pere(alain, X)      oui    X = robert  
:- grand_pere(X, robert)    oui    X = alain  
                                X = georges  
:- grand_pere(X,Y)          oui    X = alain    Y = robert  
                                X = georges    Y = robert  
                                X = georges    Y = luc
```



II - QUELQUES DEFINITIONS

Une **constante** est une chaîne de caractères commençant par une lettre minuscule, ou un nombre.

Exemples : a, sylvie, 2000

Une **variable** est une chaîne de caractères commençant par une lettre majuscule ou un _ .

Exemples : X, A, _a, Sylvie, _

Un **terme** est une constante, ou une variable, ou un symbole fonctionnel appliqué à une liste de termes.

Exemples : X, a, f(a), a+3, 3+4

Un **littéral** est un symbole de prédicat (relation) appliqué à une liste (pouvant être vide) de termes, et exprimant une propriété qui peut être vraie ou non.

Exemples : p(a), q(X), pere(alain,Y), true

Remarque: il ne doit pas y avoir de blanc entre le prédicat et la parenthèse .
