Programmation logique
Le Langage PROLOG

D. Le langage PROLOG

- **Concepts généraux**
- **Données manipulées**
- **Portée des variables et des constantes**
- **Structure d'un programme**
- **Structure interne**
- **Opérations**
- **Exemples**
- **Fonctionnement de Prolog sur un exemple**

Programmation logique
Le Langage PROLOG

Concepts généraux:

- **Créé vers les années 1970.**
- **Utilisé pour l'interrogation de base de données, Conception assisté par ordinateur, Réalisation de système expert, Compréhension du langage naturel, gestion de base de données (déductives)**
- **Particularité : Aucune distinction entre programme et données, aucune structure de contrôle.**

Programmation logique
Le Langage PROLOG

Concepts généraux:

- **Prolog implémente la règle de résolution en utilisant la technique du Backtracking(recherche en profondeur et retour arrière)**
- **Exécuter un programme revient à demander la preuve d'une expression.**
- **En programmation logique, nous pouvons trouver tous les objets en relation avec d'autres objets. (propriété de l' "invertibility")**

Programmation logique
Le Langage PROLOG

Données manipulées : Les types

- **standard : integer, real, char, symbol**
- **construit (voir les structures)**

Programmation logique
Le Langage PROLOG

Données manipulées : Objets élémentaires

— **qui peuvent être des variables et constantes.**

— **Les variables sont des combinaisons de lettres, chiffres et _. Le premier caractère est une lettre alphabétique majuscule ou '_'.**

— **Les constantes peuvent être des nombres, symbole(chaîne de caractères dont la première lettre est en minuscule, ou une chaîne de caractères quelconque entre ".**

Programmation logique
Le Langage PROLOG

Données manipulées : Listes

- **Une liste L dont les éléments sont de type type est définie comme :**

  L type*

  dans la partie "Domains".

- **[] désigne une liste vide.**
- **[a, b, c] liste formée des 3 éléments a, b et c.**
- **[X!Y] X désigne le car et Y le cdr.**

- **[a, b, c] = [a![b, c]] = [a, b![c]] = [a, b, c![]]**

Programmation logique
Le Langage PROLOG

Données manipulées : Structures

- **Utilise le mot clé Domains**

- **Exemple :**

  typedate = date(integer, integer, integer)

Programmation logique
Le Langage PROLOG

Portée des variables et des constantes

- **Les constantes ont une portée globale.**
- **La portée d'une variable se limite à une clause.**

Programmation logique
Le Langage PROLOG

Structure d'un programme :

Domains

Construction de nouveau types

Predicates

Définitions des prédicats(types des objets en relation)

Clauses

Faits et règles

Goal

But

Programmation logique
Le Langage PROLOG

Structure d'un programme :

- **:- ou if**
- **, veut dire et**
- **; veut dire ou.**

Programmation logique
Le Langage PROLOG

Structure interne :

- **La objets manipulés sont représentés en arbres.**

- **Prolog se comporte comme un démonstrateur de théorème. L'interpréteur Prolog possède donc son propre mécanisme de résolution.**

- **Il utilise les deux principes suivants :**

  - Résolution - Unification

Programmation logique
Le Langage PROLOG

Autres opérations :

- **! arrêt dés qu'une solution est trouvée(arrêt du processus du backtracking)**

- **Les règles peuvent être récursives.**

- **Malgré sa forte orientation vers la programmation logique, Prolog dispose d'un certain nombre d'opérations indispensables :**

— **communication avec l'extérieur**

— **calcul arithmétique et logique**

— **test sur la nature des variables**

— **saisie et mise à jour des règles.**

Programmation logique
Le Langage PROLOG

Exemples de programmes Prolog (Programme 1)

Predicates

Parent(symbol, symbol)

Frere(symbol, symbol)

Ascendant(symbol, symbol)

Clauses

Parent(aa, bb).

Parent(bb, cc) .

Parent(bb, dd).

Parent(dd, ee).

Parent(ff, dd).

Programmation logique
Le Langage PROLOG

```
Frere(X, Y) :- Parent(Z, X), Parent(Z, Y), Not(X=Y).
Ascendant(X,Y)  :- Parent(X, Y).
Ascendant(X,Y)  :- Parent(X, Z), Ascendant(Z,Y).

/* On peut aussi écrire ascendant comme suit :

Ascendant(X,Y)  :- Parent(X, Y);
Parent(X, Z),Ascendant(Z, Y).

*/
```

Programmation logique
Le Langage PROLOG

- **Question : frere(dd, cc)** **Réponse** **Yes**
- **Question : frere(X, dd)** **Réponse** **X = cc**
- **Question : frere(dd, X)** **Réponse** **X = cc**
- **Question : Ascendant(X, ee)**

  Réponse

  X = dd

  X = aa

  X = bb

  X = ff
- **Question : Ascendant(X, cc)**

  Réponse

  X = bb

  X = aa

Programmation logique
Le Langage PROLOG

Fonctionnement de Prolog :

1. A la question Frere(X, dd), prolog doit réfuter Non (Frere(X, dd)). Dans le cas de succès, il imprime la valeur de X.

Soit donc à réfuter D = Non (Frere(X1, dd))

Donnons quelques pas de résolution.

Frere(X1, dd) s'unifie avec Frère(X, Y)

μ = { X:=X1 ; Y:=dd }

D devient Non (Parent(Z,X1), Parent(Z, dd), X1<>dd)

Parent(Z, X1) s'unifie avec Parent(aa, bb)

μ = { Z:=aa; X1:=bb }

D devient Non (Parent(aa, dd), bb<>dd)

Comme Parent(aa, dd) est faux, ceci conduit à un échec.

Programmation logique
Le Langage PROLOG

Parent(Z, X1) s'unifie avec Parent(bb, cc)

μ = { Z:=bb; X1:=cc }

D devient Non (Parent(bb, dd), cc<>dd))

Ceci conduit à un succès avec X1 = cc.

Parent(Z, X1) s'unifie avec Parent(bb, dd)

μ = { Z:=bb; X1:=dd }

D devient Non (Parent(bb, dd), dd<>dd)

Comme dd<>dd est faux, ceci conduit à un échec.

Ect...