# SYSTEME EXPERT EN LOGO

**Michel DUPONT**

Les directives concernant l'informatique dans l'enseignement évoluent selon les aléas de la politique au gré de nos gouvernants. Pour l'heure, ceux-ci semblent surtout soucieux que le matériel soit utilisé sans avoir à faire l'effort de donner une véritable formation aux enseignants. Dans ces conditions, il n'est plus guère question d'initiation à la programmation. C'est ainsi qu'au moment où les systèmes experts deviennent réellement opérationnels dans des secteurs très variés d'activité , il n'est nulle part question d'en expliquer le fonctionnement : ni aux élèves, ni même aux enseignants. Il pourrait tout juste être utile, pour les uns ou pour les autres, de savoir en utiliser quelques uns. Les boîtes noires qu'il ne faut pas ouvrir sont de plus en plus grosses. Qu'en sera-t-il dans ces conditions de la "culture informatique" que nous sommes censés faire acquérir à nos élèves ?

Pour notre part, nous avons la conviction, que dans l'immédiat il est nécessaire qu'au moins les enseignants aient quelques connaissances en la matière. C'est dans cette perspective que, dès maintenant, nous voulons apporter notre contribution.

Nous vous proposons de commencer par saisir notre pro-gramme tel qu'il est reproduit à la fin de cet article puis de l'essayer. En voici le mode d'emploi :

> Lancez le programme en entrant : **EXPERT**
>
> Le message suivant apparaît:
> **ENTREZ LA BASE DE FAITS INITIAUX.**
>
> Si vous entrez:
> **RHIZOME FLEUR GRAINE 1.COTYLEDONE**

L'ordinateur affiche:
**MUGUET** parce que
**FLEUR** et **GRAINE** alors **PHANEROGAME**
**PHANEROGAME** et **1.COTYLEDONE** alors **MONOCOTYLEDONE**
**MONOCOTYLEDONE** et **RHIZOME** alors **MUGUET**

Cet exemple est bien connu. Il est cité dans la plupart des ouvrages qui exposent des généralités sur les systèmes experts notamment dans INTRODUCTION AUX SYSTEMES EXPERTS de Michel GONDRAN (Ed. EYROLLES).

Nous le traitons ici avec un petit programme en LOGO ce qui nous permet d'aborder ces généralités avec un exemple concret et facilement accessible. Il suffit en effet d'une connaissance minime du LOGO pour suivre nos quelques explications et comprendre le fonctionnement de notre programme.

Le moteur d'inférence est d'ordre zéro puisque le "calcul" porte sur des propositions. Cela par opposition aux moteurs d'ordre un qui peuvent utiliser des variables et des quantificateurs.

La stratégie adoptée est celle du chaînage avant puisqu'on part de faits initiaux pour démontrer d'autres faits sur lesquels l'utilisateur, supposé profane dans la discipline (ici la botanique), ne peut émettre aucune hypothèse.

A l'inverse, la stratégie du chaînage arrière supposerait que l'on parte de quelques hypothèses (les buts) qu'on tenterait ensuite de vérifier en examinant si le déclenchement en arrière des règles permet de retrouver la base de faits initiaux. Dans ce cas, les buts seraient par exemple : MUGUET, SAPIN, LILAS, ANEMONE...

Ce choix principal étant arrêté, il reste un autre choix à faire entre le fonctionnement en "profondeur d'abord" ou celui en "largeur d'abord". Nous en reparlerons ultérieurement. Nous allons en effet commencer par présenter le fonctionnement en profondeur d'abord. Nous proposerons ensuite quelques modifications à apporter au programme pour le faire fonctionner en largeur d'abord.

Mais, au préalable, nous allons examiner comment sont structurées les données.

**LES DONNEES.**

Elles sont réparties dans trois listes :

- BASEREGLES qui, bien évidemment, contient la base de règles;
- BASEFAITS qui, au départ, contient la base de faits initiaux auxquels viennent s'ajouter les faits déduits par le moteur d'inférence au fur et à mesure que des règles sont déclenchées;
- DEMARCHE: liste dans laquelle sont recopiées toutes les règles qui sont déclenchées afin que le programme les affiche en fin d'exécution pour indiquer la démarche qui a conduit au dernier fait déduit.

Qu'elle soit dans la liste BASEREGLES ou dans la liste DEMARCHE, chaque règle est structurée sous la forme d'une liste de deux objets.

* Le premier objet est lui-même une liste qui contient les faits de la partie prémisse.
* Le deuxième est un mot qui désigne le fait de la partie conclusion.

Les listes BASEREGLES et DEMARCHE sont donc des listes de listes et ces dernières listes contiennent chacune une liste et un mot.

Il faudra veiller à n'écrire chaque fait, qu'il fasse partie des prémisses d'une règle, de sa conclusion ou de la base de faits initiaux, qu'avec un seul mot. On pourra au besoin pour cela remplacer les espaces par des points (Exemple : NON.PLANTE).

**Les procédures REGLES et R.**

La structure des règles telle que nous venons de la décrire apparaît clairement dans la procédure du même nom.

A titre d'exemple indiquons que la première règle qui s'écrit R [FLEUR GRAINE] "PHANEROGAME signifie :

"si FLEUR et GRAINE alors PHANEROGAME"

ou, pour être encore plus explicite :

"si la plante possède une fleur et une graine alors la plante possède un phanérogame."

La procédure R est appelée successivement pour chacune des règles. Elle assemble les deux parties de la règle dans une liste (LISTE :PREMISSES :CONCLUSION) et elle place cette liste à la fin de la liste BASEREGLES.

Pour modifier la base de règles, il suffit donc de modifier la procédure REGLES en travaillant dans l'éditeur.

**La procédure EXPERT.**

Le programme est lancé avec EXPERT. Cette procédure effectue successivement les actions suivantes :

1. Les trois listes de données sont initialisées (listes vides).
2. Les règles sont chargées dans la base de règles (appel de la procédure REGLES).
3. La base de faits est entrée au clavier par l'exécutant.
4. Le moteur d'inférence est lancé avec la procédure MOTEUR.
5. Quand le moteur s'arrête, le dernier fait déduit est affiché à l'écran ainsi que l'ensemble de la démarche qui a permis de le déduire. Si aucune règle n'a été déclenchée, la liste DEMARCHE est alors vide et le message "aucune déduction possible" est affiché.

**Procédures LISTER1 et LISTER2.**

Ces deux procédures permettent d'afficher la démarche à la fin de l'exécution.

La procédure LISTER1 affiche ligne par ligne les règles (prémisses et conclusion) qui ont été déclenchées et qui ont donc été conservées dans la liste DEMARCHE. Notons au passage qu'en entrant LISTER1 :BASEREGLES il est possible de lister toutes les règles sous une forme plus élégante qu'en affichant la procédure REGLES.

L'affichage de toutes les prémisses d'une règle nécessite l'appel de la procédure LISTER2.

**Procédures MOTEUR et TESTPREMISSES? .**

Le moteur d'inférence proprement dit n'est constitué que des deux procédures MOTEUR et TESTPREMISSES? .

La procédure MOTEUR examine les règles les unes après les autres pour voir si elles sont déclenchables. Elle fonctionne suivant un procédé très classique en LOGO. C'est en effet une procédure récursive qui démonte la liste BASEREGLES. La règle examinée est toujours la première de la liste (PREM :REGLES). La partie de la règle contenant les prémisses est donc PREM PREM :REGLES et celle qui contient la conclusion DER PREM :REGLES.

Pour qu'une règle soit déclenchable, il faut que deux conditions soient réunies .

1. Il faut que toutes les prémisses de la règle soient dans la base de faits.

   C'est ce que vérifie la procédure TESTPREMISSES? qu'on appelle donc avec TESTPREMISSES? PREM PREM :REGLES.

   Cette procédure-prédicat fonctionne en démontant la liste des prémisses. Elle s'arrête et retourne la primitive FAUX dès qu'une des prémisses est absente de la base de faits. Si la liste est épuisée (SI VIDE? :PREMISSES), elle retourne donc la primitive VRAI.
2. Il faut que la conclusion de la règle ne soit pas déjà dans la base de faits. C'est ce qu'on vérifie avec : NON MEMBRE? DER PREM :REGLES :BASEFAITS

Quand ces deux conditions sont satisfaites et qu'en conséquence une règle est déclenchée, il s'en suit trois actions.

1. La conclusion de la règle est ajoutée à la base de faits avec DONNE "BASEFAITS MD DER PREM :REGLES :BASEFAITS
2. La régle est entrée dans la liste DEMARCHE avec DONNE "DEMARCHE MD PREM :REGLES :DEMARCHE.
3. Le moteur est relancé à partir de la première règle avec MOTEUR :BASEREGLES.

Par contre, si la régle n'est pas déclenchée, le moteur passe à la règle suivante avec MOTEUR SP :REGLES.

Lorsque la dernière règle a été examinée (SI VIDE? :REGLES) le moteur s'arrête.

**"Profondeur d'abord" et "largeur d'abord".**

Quelques modifications du programme permettent d'illustrer la différence entre ces deux stratégies : le fonctionnement en profondeur d'abord et celui en largeur d'abord.

* La "profondeur d'abord" que nous avons utilisée consiste à placer les faits déduits dans la base de faits dès qu'ils sont produits à la suite du déclenchement d'une règle.

* La "largeur d'abord" consiste à parcourir toutes les règles (C'est ce que nous appelons un cycle.) et à stocker les faits déduits au cours de ce cycle dans une liste séparée (FAITSDEDUITS) qui sera concaténée avec la base de faits à la fin du cycle. Ce travail est fait par la procédure que nous appelons tout naturellement CYCLE. Nous la donnons sans autre commentaire car, pour le reste, elle fonctionne comme la procédure MOTEUR de la version précédente.

```
POUR CYCLE :REGLES
   SI VIDE? :REGLES [STOP]
   SI ET TESTPREMISSES? PREM PREM :REGLES NON MEMBRE? DER PREM
   :REGLE :BASEFAITS [DONNE "FAITSDEDUITS MD DER PREM :REGLES
   :FAITSDEDUITS DONNE "DEMARCHE MD PREM :REGLES :DEMARCHE]
   CYCLE SP :REGLES
FIN
```

Cette opération est répétée jusqu'à ce qu'un cycle soit parcouru sans qu'aucune règle soit déclenchée. C'est ce que fait la nouvelle procédure MOTEUR.

```
POUR MOTEUR :REGLES
   DONNE "FAITSDEDUITS []
   CYCLE :REGLES
   SI NON VIDE? :FAITSDEDUITS [DONNE "BASEFAITS CONCAT :BASEFAITS
   :FAITSDEDUITS MOTEUR :REGLES]
FIN
```

Nous avons pour cela recours à une procédure de concaténation. Nous la donnons sans autre commentaire car elle est d'usage très général.

```
POUR CONCAT :LISTE1 :LISTE2
   SI VIDE? LISTE2 [RENDS LISTE1][RENDS CONCAT MD PREM :LISTE2
   LISTE1 SP :LISTE2]
FIN
```

Il ne reste plus qu'à faire fonctionner le programme avec les deux versions pour apprécier la différence et faire un choix. Vous constaterez qu'avec notre exemple cette différence est minime mais, dans les systèmes experts opérationnels, ce choix de stratégie est des plus importants.

**Version initiale.**

```
POUR REGLES
R [FLEUR GRAINE] "PHANEROGAME
R [PHANEROGAME GRAINE.NUE] "SAPIN
R [PHANEROGAME 1.COTYLEDONE] "MONOCOTYLEDONE
R [PHANEROGAME 2.COTYLEDONE] "DICOTYLEDONE
R [MONOCOTYLEDONE RHIZOME] "MUGUET
R [DICOTYLEDONE] "ANEMONE
R [MONOCOTYLEDONE NON.RHIZOME] "LILAS
R [FEUILLE FLEUR] "CRYPTOGAME
R [CRYPTOGAME NON.RACINE] "MOUSSE
R [CRYPTOGAME RACINE] "FOUGERE
R [NON.FEUILLE PLANTE] "THALLOPHYTE
R [THALLOPHYTE CHLOROPHYLLE] "ALGUE
R [THALLOPHYTE NON.CHLOROPHYLLE] "CHAMPIGNON
R [NON.FEUILLE NON.FLEUR NON.PLANTE] "COLIBACILLE
FIN


POUR R :PREMISSES :CONCLUSION
DONNE "BASEREGLES MD LISTE :PREMISSES :CONCLUSION
:BASEREGLES
FIN


POUR EXPERT
DONNE "BASEREGLES []
DONNE "BASEFAITS []
DONNE "DEMARCHE []
REGLES
VT EC [Entrez la base de faits initiaux.] EC []
DONNE "BASEFAITS LL EC []
MOTEUR :BASEREGLES
SI VIDE? :DEMARCHE [EC [Aucune déduction n'est possible.]
EC [] ] [EC PH DER :BASEFAITS [parce que] LISTER1 :DEMARCHE
EC [] ]
FIN
```

```
POUR LISTER1 :DEMARCHE
    SI VIDE? :DEMARCHE [STOP]
    LISTER2 PREM PREM :DEMARCHE EC PH [$ alors] DER PREM :DEMARCHE
    LISTER1 SP :DEMARCHE
FIN

POUR LISTER2 :PREMISSES
SI EGAL? COMPTE :PREMISSES 1 [TAPE :PREMISSES STOP]
TAPE PH PREM :PREMISSES [et$ ]
LISTER2 SP :PREMISSES
FIN

POUR MOTEUR :REGLES
    SI VIDE? :REGLES [STOP]
    SI ET TESTPREMISSES? PREM PREM :REGLES NON MEMBRE? DER PREM
    :REGLES :BASEFAITS [DONNE "BASEFAITS MD DER PREM :REGLES
    :BASEFAITS DONNE "DEMARCHE MD PREM :REGLES :DEMARCHE MOTEUR
    :BASEREGLES] [MOTEUR SP :REGLES]
FIN

POUR TESTPREMISSES? :PREMISSES
    SI VIDE? :PREMISSES [RENDS VRAI]
    SI MEMBRE? PREM :PREMISSES :BASEFAITS [RENDS TESTPREMISSES? SP
    :PREMISSES] [RENDS FAUX]
FIN
```

Michel DUPONT
AVRANCHES 1988