UN SYSTÈME EXPERT HYBRIDE
suite de l'article Hypermédias et IA paru dans le Bulletin n° 69

Gérard CLERGUE

Ce système expert a été réalisé au cours d'une recherche-action menée en commun par l'Institut National de le Jeunesse et de l'Éducation Populaire (INJEP) et le Comité de Liaison pour l'Alphabétisation et le Promotion (CLAP). Nous l'avons baptisé *JIMLOG* (jeunes immigrés et logique) pour marquer l'importance que nous accordons à la maîtrise du raisonnement et des processus de la pensée. Les droits des immigrés représentent un domaine propice à une telle étude. Dans la quête de son identité, un jeune a besoin de savoir où il en est, de connaître sa situation exacte, et ce n'est pas commode de se retrouver dans le maquis des lois et des décrets fréquemment bouleversé ces derniers temps ! Le système expert est centré sur deux problématiques : les titres de séjour nécessaires au jeune immigré résidant en France et la question de la nationalité. On envisage d'étendre ultérieurement la recherche aux problèmes du droit au travail et de la formation. Il se présente comme un outil d'aide qui fournit à chaque étape de multiples renseignements : trace du raisonnement en cours, explications sous forme d'Hypertexte, justification des déductions produites.

Des sites expérimentaux (Les Mureaux, Mantes la Jolie) vont servir de point d'appui à une première évaluation à partir de 1993. Le logiciel sera utilisé en accompagnement d'un travail d'animation mené en direction des jeunes immigrés prioritairement dans des quartiers sensibles (DSQ, ZEP...). Instrument de simulation et d'exploration il doit être le support privilégié pour amener ces jeunes à prendre conscience de la nécessité de se préoccuper de leur situation afin de pouvoir en tirer le meilleur parti. Il s'inscrit donc parfaitement dans un projet pédagogique d'autonomisation.

Les applications pédagogiques issues des Hypermédias et de l'Intelligence Artificielle ont en commun de favoriser les démarches

d'auto-apprentissage, interactives, qui laissent une grande autonomie à l'utilisateur.

Les Hypermédias sont particulièrement bien adaptés à la représentation des connaissances informelles qui utilisent des informations en clair non codées : texte en langue naturelle, images, sons. Tandis que l'Intelligence Artificielle (IA) permet de décrire les connaissances sous une forme symbolique et que l'on peut formaliser de façon rigoureuse à travers les systèmes de règles logiques grâce, par exemple, au langage Prolog (de PrologIA à Marseille) que nous avons utilisé ici pour fonctionner en tâche de fond avec Hypercard.

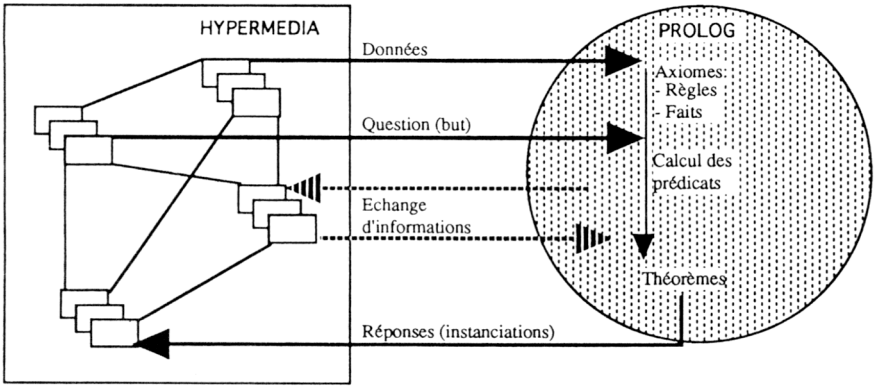

figure 1

I. LE COUPLE HYPERCARD/PROLOG

Une extension **PrologII-Hyper.INIT**, placée dans le système, assure le passage des informations entre Hypercard et Prolog. Le fonctionnement de l'ensemble est décrit dans le manuel de référence de PrologIA.

1. Du côté d'Hypercard

Sous HyperCard deux XCMD, que l'on trouve dans le fichier de ressources *HyperToProlog.XCMD.rez*, permettent d'envoyer des messages vers Prolog et d'en recevoir. Il suffit de recopier ces deux commandes dans la pile utilisée par Prolog. A tout message est associé un numéro qui peut être utilisé pour gérer la transmission.

SendProlog chaîne_de_caractères, message]

Envoie à Prolog la chaîne passée comme premier argument. Si le deuxième argument est 0, un nouveau numéro est attribué par Prolog (à partir de 10), sinon le numéro est utilisé comme numéro de message (les numéros négatifs sont réservés pour les erreurs internes).

Après l'appel, *the result* contient le numéro effectif du message envoyé, ou 0 s'il n'a pas pu être envoyé parce que le message précédent n'a pas encore été consommé. Il faut alors boucler jusqu'à ce que le message soit transmis.

GetProlog "nom_d'une_variable_HyperCard"

La procédure retourne dans *the result* la chaîne envoyée par Prolog. Si une variable est spécifiée, qui doit être déclarée comme globale, celle-ci contiendra le No du message. Si après un délai de 20 ticks (1/3 de seconde), il n'y en a toujours pas, la procédure se termine en renvoyant la chaîne "$". Il faut donc boucler jusqu'à ce que la réponse soit différente de $.

2. Du côté de Prolog

Une ressource spécifique chargée dès le départ dans Prolog est affectée à la communication avec Hypercard. Elle se trouve dans le fichier compilé *callHyperCard.PCMD.rez*.

sys:send_HC("message",n)

Le prédicat échoue si on ne peut pas envoyer le message parce que le message précédent n'a pas encore été transmis.

Il faut donc boucler avec un *repeat* jusqu'à ce que *send_HC* puisse être effacé. Dans ce cas, Si *n* est un entier > 0, Prolog envoie un message avec le numéro *n*. Si *n* = 0 ou *n* est libre, un nouveau numéro est attribué. Si *n* est libre, il est unifié avec le numéro créé.

sys:get_HC(s,n)

Le prédicat échoue s'il n'y a pas de message reçu après une attente de 20 ticks.

Il faut donc boucler avec un *repeat* jusqu'à ce que *get_HC* puisse être effacé. Dans ce cas Prolog unifie ***s*** avec la chaîne du message et ***n*** avec son numéro.

3. La vie du couple

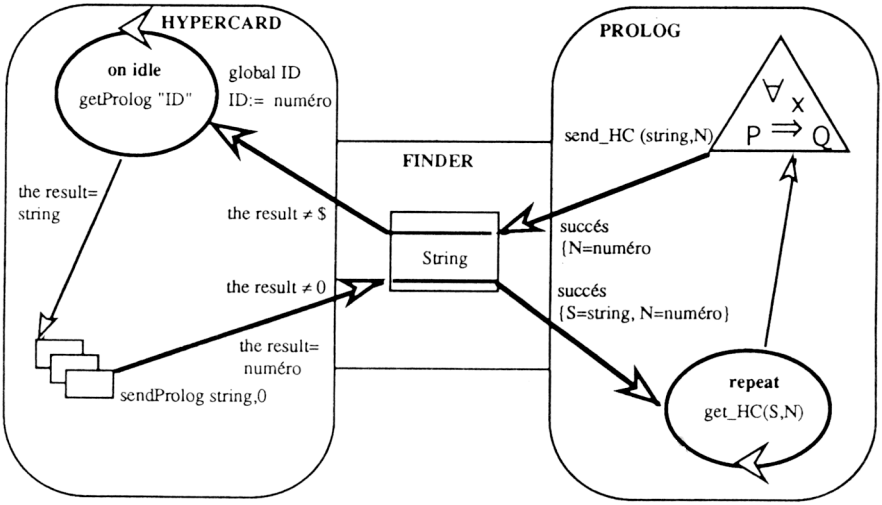

figure 2

II. UN SYSTÈME EXPERT HYPERMÉD'IA

1. L'architecture du système

Le système est constitué de deux programmes : une pile sous Hypercard et un fichier sous Prolog II+. Le lancement s'effectue à partir de la pile Hypercard qui est la plaque tournante de tout le système.

¨ On appelle le programme Prolog directement avec un bouton destiné à cet effet. Celui-ci est compilé puis Prolog se met automatiquement en attente d'un message venu d'Hypercard.

¨ On reprend la main dans Hypercard. A partir de cet instant Prolog tourne en tâche de fond de manière transparente, on peut même le faire disparaître complètement de l'écran. Toutes les opérations vont être pilotées à partir d'Hypercard, qu'il s'agisse de l'envoi des informations vers Prolog ou du traitement des informations que celui-ci a retournées.

Trois types de recherches sont proposés au départ à l'utilisateur :

1- Le séjour

Pour permettre au jeune de connaître le titre de séjour auquel il peut prétendre

2- L'éloignement

   Pour examiner toutes les procédures et les démarches à suivre en cas de situation irrégulière

3- La naturalisation

   Pour connaître les formalités à accomplir pour obtenir la nationalité française.

2. La simulation

Hypercard informe Prolog du but choisi (séjour, éloignement ou naturalisation). La démonstration peut commencer. Les règles du séjour, par exemple, attendent dans leurs prémisses des informations sur l'âge, la date d'entrée en France, le pays d'origine., etc. Toutes ces questions sont envoyées vers Hypercard qui contient au plus haut niveau dans le script de la pile une boucle perpétuelle (grâce à *on idle*) d'analyse des informations en provenance de Prolog. S'il s'agit d'une question, celle-ci sera traitée comme telle et la réponse fournie par l'utilisateur sera renvoyée vers Prolog.

Dès que la démonstration aboutit, la conclusion est dirigée vers hypercard pour y être affichée. Etant donné le non-déterminisme du langage Prolog, toutes les réponses valables sont produites. L'utilisateur aura ainsi un diagnostic complet adapté à sa situation spécifique.

A tout moment on peut connaître avec précision :

1- Le contenu de la question posée qui se présente sous une forme hypermédia
2- Pourquoi cette question a été posée, à cause de quelle règle ?
3- La trace du raisonnement. C'est à dire toutes les questions posées, les réponses que l'on a apportées et le diagnostic rendu par le système expert.

A la fin on peut demander à Prolog :

1- Le récapitulatif de tous les faits nouveaux qui ont été déduits ou apportés en cours de démonstration
2- La liste de toutes les règles qui ont pu être déclenchées

HYPERCARD	PROLOG

1
on envoyer X
{X=but lancé (séjour,…)}
{X=réponse à 1 question}
 	
sendProlog X
put the result into résultat
if résultat=0 then
envoyer X
end envoyer
	2
	Au début on lance le but repeat_get_HC
	
	repeat_get_HC ->
	repeat
	sys:**get_HC(X,N)**
	!
	traiter(X);
	
	traiter(but) -> unifier(but);

4
on idle
getProlog
put the result into résultat
if résultat ≠ $ then
traiter résultat
end idle
 	
on traiter X
if X="conclusion:…" then
afficher X
if X="question:…" then
aller à la carte X
end traiter
	3
	repeat_send_HC(X) ->
	repeat
	sys:**send_HC(X,N)**
	! ;
	
	unifier (P(A)) ->;
	P(A)
	repeat_send_HC(A)
	fail;
	unifier(_) ->;
	
	repeat->;
	repeat -> repeat;

figure 3

III. PERSPECTIVES NEUROMIMÉTIQUES

On peut encore perfectionner le système en lui faisant supporter des données bruitées ou incomplètes en entrée. Le handicap majeur de l'intelligence artificielle symbolique réside dans l'impuissance des programmes à apprendre par eux-mêmes, on est condamné à entrer préalablement toutes les connaissances qui lui seront nécessaires. Une situation nouvelle, imprévue, présentant seulement de petites différences par rapport au modèle initial est brutalement rejetée. Il en va ainsi lorsque nous ne possédons que des données floues. Il faudrait que la machine puisse accepter des informations bruitées. Par exemple une base de données que l'on veut interroger sans connaître toutes les propriétés d'un élément.

Il existe deux types de réseaux neuromimétiques : Les modèles avec apprentissage (supervisés) comme la rétropropagation et les réseaux auto-organisés comme les cartes de kohonen. Les grands domaines d'application sont :

La classification

Les réseaux peuvent servir à répartir des objets, représentés par des paramètres numériques, en plusieurs grandes classes en l'absence de critères précis. Il s'agit en fait de tranformer des données quantitatives en informations qualitatives aux contours imprécis. On rangera dans cette catégorie tous les problèmes de classification empirique et de reconnaissance des formes ou des sons.

Résolution de problèmes (exemple le problème du voyageur de commerce)

C'est le cas des réseaux auto-organisés que l'on utilise lorsqu'on ne connaît pas la solution d'avance. Sans apprentissage préalable, ils peuvent résoudre certains problèmes lorsque le réseau arrive à converger vers une solution intéressante, mais pas forcément la meilleure. En effet, le réseau peut se stabiliser dans un minimum local qui ne correspond pas à la situation optimale, il faut *l'agiter* pour qu'il cherche de lui-même un nouvel état d'équilibre plus intéressant.

Mémoire associative (exemple base de connaissances en logique floue)

L'organisation topologique des noeuds dans le réseau représente symboliquement des concepts et des relations entre ces concepts. L'activation ou non de certaines unités dessine un graphe qui peut être associé à un concept de niveau supérieur. On peut ainsi entrer des faits en activant les unités et déterminer les règles logiques souples qui les associent en jouant sur les poids des connexions. La réponse du réseau correspondra aux déductions du système.

Mac Brain est un programme qui tourne sous Hypercard et permet de réaliser cela. On pourrait grâce à lui saisir des informations nouvelles et les traiter pour obtenir avec une certaine approximation des données formelles en sortie. Celles-ci seraient alors réinjectées dans le démonstrateur Prolog sous forme de requête.

Véritable metteur en scène, Hypercard coordonne tous les acteurs de la pièce où les outils d'intelligence artificielle se produisent dans un environnement hypermédia.

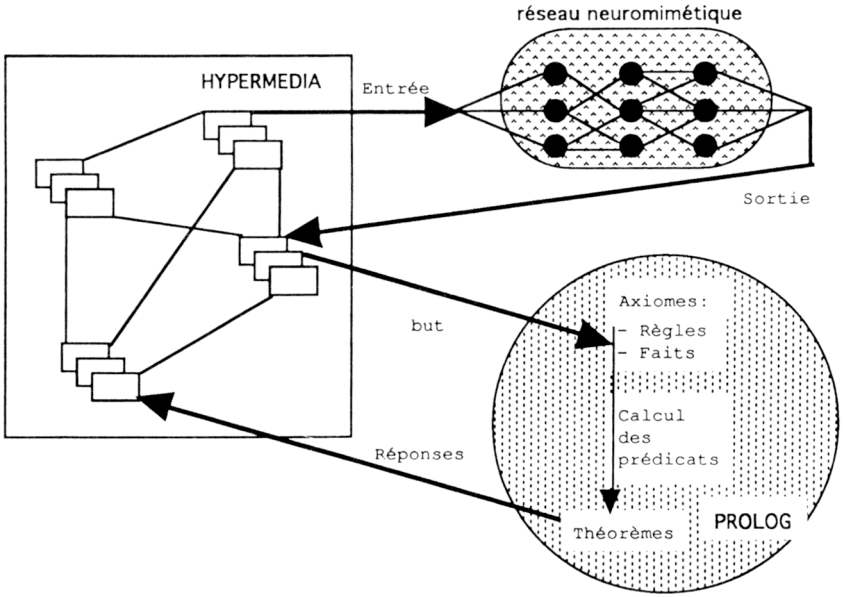

figure 4

CONCLUSIONS

1- Un hypermédia (HyperCard) pilote toutes les opérations, c'est le coeur d'une station de travail pour développer des stratégies d'auto-apprentissage. D'autres modules peuvent encore venir s'ajouter comme : le traitement des données bruitées grâce à un réseau neuromimétique

2- Navigation sous Hypercard par association d'idées

3- Démonstration non-déterministe de Prolog en tâche de fond, complètement transparente

4- Prise en compte de la complexité dans l'acte cognitif qui intègre le symbolique et l'analogique.

Gérard CLERGUE

Responsable du laboratoire informatique de l'INJEP à Marly

Chargé de cours en Sciences de l'Education à l'Université de Paris X

BIBLIOGRAPHIE

D. HOFSTADTER, *Eveil après le rêve booléen ("Ma Thémagie")*, Inter-Editions, 1982)

P. LÉVY, *Les technologies de l'intelligence*, La découverte, 1990.

F. LÉONARD, G. SACKUR (1991) *Connaissances locales triple approche, une méthodologie de recherche*, Revue de didactique des mathématiques, 10, 205-240.

A. GRUMBACH, *Modèle symboliconnexionniste de l'apprentissage*, LIFIA-IMAG,ENST, 1990.

PrologIA, *Manuel de référence de Prolog II+ et Prolog III*, Luminy, Marseille.

G. CLERGUE, *Usages et langages de l'intelligence artificielle*, INJEP, 1990.

J.P. BUISSON, G. CLERGUE, *Le connexionnisme : les réseaux de neurones*, INJEP, 1992.