DÉDUCTION AUTOMATIQUE ET SYSTÈMES TRANSFORMATIONNELS

J. CHAUCHÉ

C.E.L.T.A. 23, Boulevard Albert 1er
54000 - NANCY FRANCE

RESUME :
Les systèmes transformationnels utilisent des processus déductifs d'une approche différente des systèmes utilisés en intelligence artificielle. A travers une comparaison du langage Prolog et du langage Sygmart, il est montré comment réaliser dans les systèmes transformationnels des applications utilisant des raisonnements et des bases de connaissances.

I. INTRODUCTION : Le traitement algorithmique d'un texte sur ordinateur suppose une approche formelle, une projection du texte dans cette approche et enfin un traitement du texte ainsi projeté. L'approche la plus courante à l'heure actuelle en intelligence artificielle est celle de la démonstration automatique dans un système logique. Cette approche est procédurale, c'est-à-dire qu'elle traite avant tout la forme prédicative du langage. Du point de vue formel les objets manipulés sont réduits à des éléments simples: atomes ou assemblage d'atomes suivant des règles de formation de prédicat. L'approche définie dans un système transformationnel est avant tout orientée objet. Elle se caractérise par la définition d'un univers de traitement et d'un processus de manipulation des objets de cet univers. Les problèmes classiques de déduction ou d'inférence se transposent naturellement dans ce type de manipulation. Ainsi l'approche structurelle du traitement des langues naturelles apporte une vision globale : elle permet une analyse et une exploitation de cette analyse. Elle peut également servir d'outil à la formalisation de la représentation des connaissances. Le but de cet article est de montrer à travers quelques exemples simples la transposition transformationnelle des principales fonctions déductives des systèmes bases sur l'inférence logique. Les exemples utilisés sont ceux définis dans PROLOG [3] pour l'approche logique et sont traduits dans SYGMART [2] pour l'approche transformationnelle.

2. OBJET ET REPRESENTATIONS :
Deux modes de représentations sont utilisés pour le traitement des langues. La représentation prédicative comprend : des atomes, des variables et des prédicats. La représentation structurelle comporte trois éléments : structure, étiquette, fonction d'étiquetage :

- Une structure est un ensemble d'arborescences sur un ensemble de points.
- Une étiquette est un ensemble de variables affectées de valeurs
- Une fonction d'étiquetage associe à chaque point une étiquette de l'ensemble des étiquettes de la structure.

Une structure peut être le résultat d'une analyse ou l'entrée d'un processus transformationnel.

La nature des objets manipulés se distingue donc nettement en fonction de l'approche définie. Les étiquettes dans le système SYGMART par exemple, recouvrent un ensemble important de variables et les principales propriétés associées à un élément seront réunies dans ce système dans l'étiquette associée à cet élément.

L'équivalent dans le système PROLOG nécessite toute une structure et un ensemble de règles d'accès aux valeurs correspondantes.

Exemple : la phrase "the man eats the apple" a comme structure dans l'application [3] :
sentence (noun-phrase(determiner(the), noun(man)), verb phrase(verb(eats), noun-phrase (determiner(the), noun(apple)))).

La même structure et l'application des règles transposées donnerait dans le système SYGMART beaucoup plus d'information :
PH(GN-Sujet(déterminant-masculin-singulier-the, nom-masculin-singulier-man), GV-actif(verbe-présent-3e personne-singulier-eats, GN-complément-objet-direct (déterminant-neutre-singulier-the, nom-neutre-singulier-apple))).

Evidemment comme il est précisé plus haut, la prise en compte dans PROLOG de tous ces renseignements suppose une structure beaucoup plus complexe.

3. ALGORITHMES TRANSFORMATIONNELS ET REPRESENTATIONS.

La nature de la représentation recherchée influe sur l'algorithme utilisé et la connaissance algorithmique limite les formes possibles de la représentation. L'analyse effectuée par un système est donc très dépendante des choix déterminés pour la représentation interne qui est le but de cette analyse. Les choix de cette représentation interne sont eux mêmes très dépendants des algorithmes définis pour l'analyse.

L'analyse dans un système déductionnel comportera une notion de globalité de l'analyse. A chaque instant l'analyse recherchera un découpage de la phrase mise sous forme de liste. Ce découpage devra être en accord avec les règles de déduction du système. Ainsi dans [3] l'analyse est basée sur le prédicat "append" de fusion de liste : append (X,Y,Z) est vrai si Z est la liste fusion des listes X et Y et l'analyse de la phrase s'écrit :
sentence(X): -append(Y,Z,X),noun-phrase(Y),verb-phrase(Z). Soit [The man eats the apple] → [[the man]_X [eats the apple]_Z]_Y

Dans cette approche une analyse partielle ne peut aboutir car chaque prédicat associé à append doit être vérifié. Le résultat de cette analyse est également très spécifique et constitue une arborescence binaire simplement étiquetée.

Dans un système transformationnel l'approche est locale par opposition à l'approche globale précédente. Les transformations seront toutes locales et il est nécessaire de prévoir un test de réalisation si une analyse est exigée. La règle PROLOG de construction de la structure d'une phrase est [3] :

```
sentence (X,sentence(NP,VP),SO,S):-
    noun-phrase(X,NP,SO,S1),
    verb-phrase(X,VP,S1,S).
```

La règle transformationnelle correspondante est la suivante :

O(*,NP,*,VP,*) ⇒ O(sentence(NP,VP)).

La grammaire transformationnelle strictement équivalente à la grammaire définie dans [3] est la suivante :

O(the) ⇒ O(determiner(the)).
O(man) ⇒ O(noun(man)).
O(apple) ⇒ O(noun(apple)).
O(eats) ⇒ O(verb(eats)).
O(déterminer,*,noun) ⇒ O(noun-phrase(déterminer, noun)).
O(verb,*,noun-phrase) ⇒ O(verb-phrase(verb,noun-phrase)).
O(noun-phrase, *,verb-phrase) ⇒ O(sentence(noun-phrase,verb-phrase)).

Le résultat est alors identique.

L'analyse s'effectue ici de bas en haut : reconnaissance des éléments simples, puis des groupes et enfin de la phrase. Dans le cas où la phrase d'entrée ne serait pas correcte l'analyse donnerait un résultat contrairement à la grammaire PROLOG.
Si par exemple la reconnaissance de "eats" est défectueuse nous aurons le résultat :

O(noun-phrase(déterminer(the),noun(man)),X,noun-phrase (déterminer(the),noun(apple))).

Pour une équivalence stricte des deux grammaires il est donc nécessaire dans la grammaire transformationnelle d'ajouter un test de satisfaction. Ce text pour l'exemple précédent est défini par la présence du schéma suivant dans le résultat :

O(*,sentence,*).
ou de la définition de la règle
O(*,sentence,*) ⇒ sentence.

4. ALGORITHMES TRANSFORMATIONNELS ET BASES DE CONNAISSANCES.

Dans les systèmes d'intelligence artificielle l'utilisation des connaissances est essentielle. L'ensemble des connaissances est utilisé pour déduire de nouvelles connaissances ou de nouvelles propriétés. Cette déduction simule une déduction logique et il n'y a pas de différence fondamentale entre la déduction et le traitement transformationnel. Ainsi une règle transformationnelle A ⇒ B signifie que l'on peut déduire l'état B à partir de l'état A. La transposition d'un exemple PROLOG [3] apporte une illustration de ce problème :

Soit la base de fait :
parent(marie,jean,yvette).
parent(claude,jean,yvette).
femme(marie).
homme(claude).
et soit la règle :
soeur(X,Y) :- femme(X),parent(X,M,F),parent(Y,M,F).
A la question :
soeur(X,claude) PROLOG répond : marie.

L'équivalent dans un système de remplacement est le suivant :
Règle :
soeur(X) ⇒ fille(A,B).
règle définissant un nouveau prédicat et nécessitant une lecture de la base de connaissance qui affecte a A et B le nom des parents.

fille(A,B) ⇒ Y
règle de consultation de la base de connaissance.
Y est obtenu par consultation de cette base.

La base de connaissance est ici un dictionnaire d'étiquettes. Plusieurs choix sont bien sûr possibles. Définissons-en un associant à chaque entrée quatre variables : nom,père,mère,sexe. L'entrée ainsi définie comportera éventuellement des renseignements supplémentaires. La base sera alors la suivante :
marie,F,jean,yvette :marie,F,jean,yvette,ne le ...,...,...
claude,M,jean,yvette:claude,M,jean,yvette,ne le ...,...,...

La consultation de la base s'effectue suivant la clef nom ou la clef complexe sexe,père,mère. Ainsi la première règle trouvera la première ligne et la deuxième règle la deuxième ligne. A chaque consultation les renseignements associés sont complexes et composés de toutes les variables associées à l'entrée. Dans le cas de cet exemple lès règles n'ont pas à effectuer nécessairement une transformation puisque dans ce cas la consultation permet à elle seule de définir cette déduction. Les deux règles sont donc les suivantes :

X ⇒ X avec comme condition : la variable "pred" doit être égale à "soeur" et la consultation du dictionnaire doit définir un résultat. L'étiquette associée à X sera modifiée, la variable "pred" prendra la valeur "fille" la variable "sexe" prendra la valeur "F" et les variables "père" et "mère" seront affectées des valeurs lues dans le dictionnaire avec comme clé : "nom".

X ⇒ X avec comme condition : la variable "pred" doit être égale à "fille" et la consultation du dictionnaire doit définir un résultat.

L'étiquette associée à X sera modifiée, la variable "nom" prendra la valeur lue dans le dictionnaire avec comme clé : sexe, père, mère.

Il y a une opposition entre complexité d'une clé et nombre de transformations test à réaliser. Ainsi avec une clé simple de la forme :

nom : père,mère,ne le, etc...

la discrimination sur le choix de l'entrée s'effectuera une fois la substitution, c'est-à-dire la règle effectuée. Alors qu'avec la clé complexe définie précédemment le choix sera quasiment déterministe.
La consultation du dictionnaire dans un système transformationnel transpose la substitution. A une arborescence quelconque une règle associe une valeur d'étiquette complexe. Cette étiquette permet elle-même de définir une entrée dans un dictionnaire. L'entrée ainsi définie détermine complètement la transformation qui devient dépendante de la base de connaissance.

Un système transformationnel est avant tout déterministe. Le cas de l'essai sur plusieurs solutions doit être défini par la grammaire elle-même. Plusieurs maîtrises sont possibles et dépendent bien souvent du cas considéré. Prenons l'exemple le plus simple et le plus général qui définit l'énumération des choix sans considération particulière. Soit la variable n définissant le numéro de choix (cette variable arithmétique est mise à jour à la construction du dictionnaire ou à sa modification). Il suffit alors d'ajouter une règle de choix aux deux règles précédentes :

X ⇒ X choix de la solution suivante : n est augmenté de un.

Les règles sont ordonnées donc après l'application de R2 la règle R3 est applicable et va permettre la consultation du dictionnaire avec le choix suivant. Bien sûr le contrôle et l'exploitation de ces diffé-

rents choix appartient éventuellement à d'autres règles. A l'inverse d'une recherche combinatoire qu'il faut maîtriser par des fonctions spécifiques (exemple de "cut" en PROLOG) l'approche transformationnelle est déterministe et les problèmes de choix éventuels doivent être explicitement décrits et contrôlés.

5. ALGORITHMES TRANSFORMATIONNELS ET OBJET MANIPULES.

Un système de remplacement opère par transformation locale d'un objet. Contrairement à un système déductif qui impose une forme d'objet dans les systèmes de remplacement, peuvent être construits sur des classes d'objets très divers. Il suffit pour un tel système de définir deux éléments : l'univers des objets et les opérateurs de transformations. Ainsi les déductions opèreront sur des objets non triviaux et seront déjà d'une complexité bien supérieure à la déduction prédicative. Comme exemple à ce mode de manipulation considérons la transformation "maplist" définie dans [3]. Cette transformation ne s'effectue que par lecture complète de la phrase et est définie par les règles suivantes :

```
change(you,i).
change(are,[am,not]).
change(X,X).
maplist(_,[],[]).
maplist(P,[X|L],[Y|M]) :-
  Q =.. [P,X,Y],
  call Q,
  maplist(P,L,M).
```

et maplist(change,[your,are,a,computer],Z).
determine Z = [i,[am,not],a,computer].

La définition transformationnelle de cette phrase est immédiate :

PH(GN(you),GV(are)) ⇒ PH(GN(i),GV(GVN(are,not))).

Cette règle transforme toute phrase contenant "... your are ..." en une phrase de la forme "... i am not ...".

En fait l'approche transformationnelle sera plus complexe sur le traitement des étiquettes et sera écrite de façon à définir une réponse négative quelque soit le verbe utilisé :

PH(GN(pronom),GV(VB)) ⇒ PH(GN("équivalent(pronom)"),GV(VB-nég)).

avec comme équivalent : 2ème personne → 1ère personne
I am a computer → i am not a computer.
we are a computer → we are not a computer.
your are a computer → i am not a computer.

mais aussi :

you like a computer → i do not like a computer.

car la forme négative du verbe est traitée au moment de la génération de la phrase.
La structure manipulée dans un système transformationnel est souvent plus complexe et n'a pas d'équivalent simple dans un système déductif. En effet dans un système déductif chaque renseignement complémentaire associé à un élément (mot du texte par exemple) doit être défini par une variable ou un atome. Cette contrainte implique un double rôle à la structure traitée : un rôle semantique impliquant les relations entre les différents éléments du texte et un rôle opératoire nécessaire pour la manipulation des renseignements associés. En définissant des transformations d'objets complexes les systèmes transformationnels utilisent seulement la première relation, ce qui élimine les règles opératoires. Dans le système SYGMART les objets manipulés sont plus complexes que des structures simples comme des listes ou des arborescences et n'ont pas d'équivalent naturels dans les systèmes déductifs.

6. SYSTEMES TRANSFORMATIONNELS ET ALGORITHMES.

L'expression d'un algorithme dans un système transformationnel suppose une définition préalable de la représentation des éléments traités par cet algorithme. Le problème classique du tri par exemple s'exprime très simplement de façon transformationnelle mais on doit définir ce que l'on appelle objet trié. Dans une représentation arborescence on peut définir un objet trié une arborescence ordonnée dont tous les descendants d'un point sont triés. C'est-à-dire que si les points A et B dépendent tous les deux de la racine et si A précède B dans l'ordre des descendants alors on a f(A) < f(B) (f étant la fonction d'étiquetage des points). Dans cette convention l'algorithme classique du "bubble sort" s'écrit :

O(A,B) / f(A) > f(B) ⇒ O(B,A).

Les différentes transformations simuleront les permutations associées à ce tri.
Un autre exemple académique est celui des "tours de Hanoi". Il est nécessaire au préalable de définir le but de cet algorithme. Le plus simple est la recherche de la structure définissant tous les mouvements. La lecture canonique de cette structure donnera le résultat.

La forme générale de cette structure est la suivante :

O(X,M(Y,Z),T) ou X et T sont les racines des structures définissant des mouvements et M(Y,Z) définit un mouvement de Y vers Z. Une étiquette est associée à chaque point et contient soit le nom d'une tour, soit le nombre de disque concerné. L'algorithme s'exprime alors en une règle :

```
O(A,B,C) / O: n > 0 ⇒O(X(A,C,B),M(A,B),Z(C,B,A)) /
O : n = 0      X:n = n(O) - 1 ; M: n=0 ;
               Z:n = n(O) = 1.
```

Un système transformationnel suppose deux fonctions externes. La première permet de projeter les éléments à traiter dans les objets manipulés par le système et la seconde définit l'opération inverse. Ainsi dans le cas des tours de Hanoi la première fonction projetera un nombre sur une structure ayant quatre éléments :

O(I,J,K).

Le nombre initial sera affecté à la variable n de l'étiquette associée à O ; et les noms des tours I,J,K seront affectés aux étiquettes associées aux points I,J,K. Sur ces étiquettes la variable n sera bien sûr nulle. La fonction de visualisation imprimera le nom des tours sur les points qui ont un nom et le symbole "→" par exemple sur les points M. En général ces fonctions sont plus complexes et dépendent de l'application traitée. Dans le système SYGMART ces fonctions sont associées à un traitement morphologique (analyse ou synthèse morphologique) de façon à travailler sur des textes en langues naturelles.

7. BIBLIOGRAPHIE.

[1] Chauché, J. Transducteurs et arborescences. Thèse d'état Grenoble 1974

[2] Chauché, J. Un outil multidimensionnel de l'analyse du discours Proceeding of Coling 84. Standford University, California 1984.

[3] Clocksin W.F. & Mellish C.S, Programming in Prolog. Springer-Verlag 1981,

[4] Mendelson E. Introduction to Mathematical Logic Van Nostrand Reinhold 1964

[5] Rolf, P.C. Vertalen van getalsnamen, Verslagen Computerlinguistiek N. 3 KU Nijmegen 1983

[6] Rolf P.C. & Chauché J. Machine translation and the SYGMART system. Computers and the Humanities à paraître

[7] Vigroux Ch. Bilan d'une expérience en traduction automatique Espagnol Français Anglais. Cahier du CRAL n° 42 - NANCY

[8] Wang H. La place de la modalité dans un système de traduction automatique trilingue Français-Anglais-Chinois. Thèse Nancy 1983.

-=-=-=-=-=-=-=-=-=-