TRAVAUX DU CENTRE DE RECHERCHES SÉMIOLOGIQUES

Département Information et
Formation en Economie
CNRS - IRPEACS - Lyon

Centre d'Analyse du Changement
Economique et Social. Département
du C.A.M.S.
CNRS - EHESS - Marseille

Centre de Recherches Sémiologiques
Université de Neuchâtel

NOUVELLES TECHNOLOGIES ET REPRÉSENTATIONS DE SALARIÉS FRANÇAIS ET SUISSES

Document de travail

Analyse de texte assistée par ordinateur
Utilisation du logiciel DEREDEC

No 48 - janvier 1985

CdRS

NOUVELLES TECHNOLOGIES ET REPRÉSENTATIONS DE

SALARIÉS FRANÇAIS ET SUISSES

Document de travail

ANALYSE DE TEXTE ASSISTEE PAR ORDINATEUR

UTILISATION DU LOGICIEL DÉREDEC

NO 48 - JANVIER 1985

Table des matières

LA STRUCTURE DES DONNEES ET DES ALGORITHMES EN DEREDEC

Pierre Plante

Table des matières

** Le présent document est en très grande partie composé d'extraits du Manuel de programmation du Déredec (Version 1.0, éditée par SAPIA, Montréal 1984).

AVERTISSEMENT

Ce cahier constitue un premier document de travail qui s'inscrit dans le cadre d'une recherche intitulée

"Nouvelles technologies et représentations de salariés français et suisses".

Trois institutions y travaillent en collaboration:

A Lyon: le Département Information et formation en économie (A. Silem), CNRS-IRPEACS (directeur J.-M. Albertini);

A Marseille: le Centre d'Analyse du Changement économique et social du Département C.A.M.S. (P. Vergès), CNRS-EHESS;

A Neuchâtel: le Centre de Recherches sémiologiques (J.-B. Grize), FNSRS*.

Il était indispensable, au vu du nombre des entretiens réalisés et de leur complexité, de disposer d'une aide informatique pour les analyser. Les programmes nécessaires ne sont pas nombreux et nous ne pouvions songer à en créer un. Grâce à l'amitié de Pierre Plante, nous avons pu utiliser le logiciel DEREDEC, dont il est l'auteur.

Restait à le "faire tourner", ce qui n'était possible qu'à Montréal. Cette opération aurait dépassé -et de loin- les moyens financiers dont nous disposions sans la générosité et l'amabilité du Centre d'Analyse du Texte par ordinateur (directeur J. Duchatel) de l'Université du Québec à Montréal (UQAM).

Au nom des trois groupes de recherche, de leurs responsables et de leurs collaborateurs, je tiens à exprimer ici toute notre reconnaissance. Je me réjouis aussi de ce lien avec l'UQAM, qui vient s'ajouter à d'autres que nous avions déjà.

---

* Requête no 1'743-0.83.

Le lecteur trouvera dans les pages qui suivent, d'une part la présentation par son auteur du logiciel DEREDEC, d'autre part, trois exemples de son utilisation. Ils sont dus à ceux qui utilisent depuis plusieurs années en Europe le programme DEREDEC -le GRAD de l'Université de Grenoble I- et l'un d'entre eux prend appui sur une partie du corpus qui est le nôtre.

Jean-Blaise GRIZE

Lyon, Marseille, Neuchâtel
janvier 1985

LA STRUCTURE DES DONNEES ET DES ALGORITHMES EN DEREDEC

Pierre Plante

1. IMPLEMENTATIONS

Le Déredec est un logiciel actuellement opérationnel en NEW-UCI LISP (sur des machines DEC-10), en INTERLISP (sur IBM, en version réduite) et en IQ-LISP sur micro-ordinateur IBM PC.

Une version pour VAX (Digital Equipment Corporation) en LISP NIL (M.I.T.) sera bientôt disponible.

2. VUE D'ENSEMBLE

Le Déredec est un logiciel de traitement linguistique, d'analyse de contenu des textes et de mise au point de systèmes-experts en langage naturel. Il s'agit d'un système de programmation exigeant peu de connaissances préalables en informatique. Tout chercheur en sciences humaines (philosophie, linguistique, psychologie, sociologie...) peut ainsi, à l'aide de cet outil, mettre au point et appliquer ses propres hypothèses de description de texte (description morphologique, syntaxique, sémantique, logique...), ses propres idées sur l'exploration des descriptions construites et l'analyse du contenu des résultats, ainsi qu'éventuellement son propre système de questions/réponses.

Le Déredec se définit comme un cadre computationnel général, simple et souple pour le traitement polyvalent des langues naturelles.

Le Déredec facilite la programmation d'analyses situées à d'autres niveaux que ceux auxquels nous ont habitués les logiciels con-

sacrés au dépistage des concordances et à l'analyse des cooccurrences.

Dans la production des concordances (segments d'un texte renfermant un mot clé donné), et dans l'analyse des cooccurrences (deux mots ont une relation de cooccurrence s'ils appartiennent au même segment), le segment, c'est-à-dire l'unité d'investigation contextuelle se définit habituellement dans les termes de certains traits d'édition (par exemple la "ligne" ou la "page") ou de bornes numériques (un certain nombre de mots avant ou après le mot étudié). Par la suite l'analyseur opère des regroupements statistiques sur les mots des segments dépistés.

Quelle que soit la sophistication des calculs effectués sur ces regroupements, l'interprétation finale des résultats reste toujours liée à la pauvreté initiale de l'investigation contextuelle.

C'est ce constat qui a motivé le développement du Déredec. Nous voulions répondre au besoin d'un logiciel général consacré à la programmation d'investigations contextuelles définies dans les termes d'une composition structurale dont le niveau de complexité soit arbitraire et dont le contenu puisse être associé à la solution de différents problèmes linguistiques.

Alors que dans l'analyse classique des cooccurrences, toutes les relations entre les mots dans une phrase sont aplaties à la relation de covoisinage, les analyseurs Déredec permettront, au moyen d'algorithmes d'indexation de symboles descripteurs, de privilégier certaines relations dans les phrases.

Le Déredec va permettre la programmation d'algorithmes où la notion de contexte peut se définir autrement que par des traits d'édition, ou par un simple décompte opéré sur la suite des mots du texte. En fait, la structure des expressions admissibles dans laquelle l'utilisateur sera amené à définir ses unités d'investigation est suffisamment sophistiquée pour traiter plusieurs des problèmes qui caractérisent la simulation des phénomènes de compréhension des langues naturelles: divers types de désambiguïsation syntaxico-sémantique, les simulations de dialogues et de processus cognitifs, la production de paraphrases, etc.

Les appellations "texte" et "description de texte" renvoient aux objets privilégiés d'analyse Déredec. On notera que le logiciel peut en fait s'appliquer à tous les objets ayant comme les textes la forme de séries d'événements et dont on imagine que les descripteurs

puissent avantageusement se structurer en arborescences. Retenons donc sous l'appellation "texte" l'idée très générale d'une concaténation d'éléments, ceux-ci pouvant être des mots dans des phrases, mais aussi des entités aussi diverses que des comportements dans une durée, des véhicules sur une route, des notes de musique sur une portée...

Le Déredec est un produit LISP. De l'intérieur du logiciel, l'usager a toujours accès aux fonctions de base de ce langage, à son éditeur et aux différentes structures de trace et de mise au point. Un usager averti peut ainsi augmenter à sa guise le Déredec de procédures particulières ou encore regrouper dans de nouvelles fonctions LISP des routines Déredec régulièrement appelées. Par ailleurs la connaissance de LISP n'est pas requise à l'utilisation stricte des fonctions Déredec.

Le Déredec est un système de programmation à trois volets. On y trouve:

a) Une structure de représentation (stockage) des données

Les "objets textuels" tels qu'on les identifie en début d'analyse ou tels qu'on les retrouve à la suite de manipulations linguistiques, possèdent toujours la même structure de formation. Celle-ci se nomme EXFAD pour EXpressions de Forme Admissible aux Descriptions. On appellera DDT (Description de Texte) toute suite d'EXFAD.

Tous les objets textuels (mots, phrases, suite de phrases, textes, etc.) dès qu'ils sont déclarés admissibles au Déredec, jusqu'à la sortie des programmes de l'usager ayant permis de les transformer en structures descriptives, sont toujours et uniquement des EXFAD. Une EXFAD peut donc être un objet très simple tel un mot, mais aussi un objet plus complexe tel un réseau sémantique ou une phrase munie de sa structure syntaxique.

C'est cette caractéristique d'unicité dans la formule de représentation des données qui rend possible en Déredec la programmation sans interfaces entre elles de machines consacrées à l'obtention de structures linguistiques variées: morphologique, sémantique, syntaxique, pragmatique...

Des Dictionnaires (de catégories ou de réseaux sémantiques) et des Lexiques (listes fréquentielles diverses) servent aussi en Déredec à stocker l'information.

b) Une batterie de fonctions pour la manipulation des données

- Des fonctions descriptives ou fonctions de constructions des EXFAD

Les suites d'EXFAD (ou DDT) sont obtenues par la programmation (par l'usager) d'automates à états finis spéciaux (subséquemment appelés automates Déredec). Ces automates analysent par un balayage contextuel les séquences d'EXFAD déposées sur un fichier d'entrée, et construisent sur celles-ci de nouvelles EXFAD descriptives des régularités linguistiques observées.

- Des fonctions exploratrices

Les DDT produites par les automates seront analysées par des fonctions exploratrices dont les arguments (appelés modèles d'exploration), fournis par l'usager, sont des structures de "pattern matching" ayant une syntaxe d'écriture simple et un pouvoir de discernement élevé. C'est par le biais des fonctions exploratrices que les grammaires descriptives de texte seront associées à des objectifs d'analyse de contenu.

c) Une structure de communication entre les volets a et b

Des fonctions Déredec transforment de différentes façons les structures de données en procédures de manipulation et certaines procédures de manipulation en structures de données. En fait, certaines entités Déredec n'ont pas de vocation définitive et peuvent, dépendant du point de vue où elles sont observées être tantôt considérées comme des données, tantôt comme des programmes.

Dans un scénario simple d'utilisation du logiciel, les fonctions du volet c ne seraient pas appliquées.

Imaginons un chercheur intéressé à comparer pour cinq textes différents tous les adjectifs situés dans les phrases en position de déterminants nominaux (la grosse pomme, les grands arbres, la table noire, etc.). Dans un premier temps il entrera ses textes sous autant de fichiers, à l'aide d'un simple éditeur de texte. Puis il programmera sa Grammaire de Texte (GDT). On appelle ainsi un jeu d'automates à états finis Déredec susceptibles de plaquer sur les séquences d'entrée les structures descriptives désirées. Les automates Déredec sont programmés par l'usager pour effectuer des balayages contextuels sur les éléments

des séquences et y laisser différentes traces susceptibles d'être explorées par la suite.

Une fois sa grammaire programmée, l'usager se trouve habituellement aux prises avec les différentes difficultés d'application de celle-ci sur son corpus.

Ici le logiciel lui porte assistance de plusieurs façons: d'abord il dépiste et diagnostique pour lui, et de façon automatique, des erreurs de programmation, il permet aussi de suivre à la trace le comportement d'un automate, de même il autorise des interruptions dans le travail des automates afin de questionner l'état de la description, de modifier la grammaire...

A la fin de ces séances de mise au point, l'usager aura obtenu ses DDT. Dans notre exemple, cela signifie que toutes les relations de déterminations adjectifs-nominaux auront été plaquées sur les séquences d'entrée des cinq textes.

La prochaine étape du projet porte normalement sur l'exploration de ces résultats. L'usager s'adonne alors à la programmation de modèles d'exploration. Celui de notre exemple pourrait par un modèle d'exploration rassembler tous les nominaux déterminés par une classe d'adjectifs d'abord regroupés sous une même catégorie, etc.

Puis suivent les comparaisons intertextuelles. Sur les différents fichiers contenant ces résultats d'exploration, notre usager commandera l'exécution de comparaisons diverses par l'entremise de fonctions d'analyse, afin d'étudier les différences de comportement de ses cinq textes eu égard à cette relation de détermination adjectivale.

Les séances de programmation en Déredec auront ainsi habituellement l'aspect d'un enchaînement de constructions d'automates, d'obtentions de DDT par l'application de ces automates sur le corpus, puis d'élaborations de modèles d'exploration, d'analyses des résultats éventuellement suivies de réexplorations ou de reconstructions de nouvelles descriptions. Programmer en Déredec signifie essentiellement programmer la production de DDT, puis programmer l'exploration de ces dernières, analyser les résultats et recommencer à l'une ou l'autre des étapes jusqu'à ce que ces résultats soient satisfaisants. L'ensemble de la démarche est hautement facilité par le fait que les expressions admissibles à l'entrée comme à la sortie, à la fois des fonctions descriptives et des fonctions ex-

ploratrices, ont, du point de vue informatique, exactement la même syntaxe d'écriture.

On peut penser que ce type d'expérimentation sera le lot de la majorité des usagers du logiciel. Mais l'usager très intéressé voudra sûrement goûter aux procédures PASC du Déredec, ces procédures de Programmation Automatique Sensibles au Contexte.

Il s'agit de retirer des mains du programmeur la majorité des opérations à effectuer au cours d'une séance de programmation, pour ne lui laisser que certaines décisions de haut niveau relatives à la planification générale des expériences.

Par exemple, certaines procédures PASC concernent la réapplication en chaîne de fonctions exploratrices sur une DDT; les résultats obtenus à chaque exploration servent à modifier le ou les modèles d'exploration utilisés, modèles dont une première version est donnée au point de départ par l'usager. Ce dernier contrôle la procédure PASC en fournissant certaines clés, certains paramètres qui guideront l'ensemble des opérations.

D'autres procédures PASC permettent d'enrichir progressivement une DDT en modifiant constamment un apparatus descriptif dont la structure générale est fournie au début, accompagnée là aussi de paramètres généraux sur la conduite du processus récursif.

On notera que les procédures de programmation automatique ont du point de vue de leur syntaxe informatique la même forme ou la même admissibilité que les autres fonctions ou opérations Déredec, de telle sorte qu'elles sont composables avec ces dernières. C'est cette caractéristique qui rend compte de l'étiquette "sensibles au contexte" dans l'appellation PASC: les procédures de programmation automatique sont logées dans des environnements susceptibles de fournir les paramètres pertinents à leur propre exécution.

La machinerie PASC qui de loin est la plus complexe et en quelque sorte la plus complète a pour nom APLEC.

APLEC (APprenti-LECteur) associe de façon automatique un module de questions/réponses pour toute GDT soumise par un usager, module où les questions sont formulées dans la langue naturelle du texte et où les réponses sont des segments dépistés dans celui-ci.

Lorsque APLEC ne peut pas dépister de réponse, il diagnostique la difficulté, intervient auprès de l'usager, s'enquiert d'une solu-

tion et tente de la généraliser pour tout le corpus, ceci, afin d'augmenter son pouvoir de fouille ultérieur et d'éviter le plus possible les interruptions.

En plus de la grammaire et de quelques modèles d'exploration (qu'il considère comme représentatifs des structures indexées par celle-ci), l'usager ne fournit à APLEC que des paramètres très généraux sur les conditions de fouille et les conditions d'apprentissage.

APLEC permet de distinguer formellement ce qui dans une entreprise sémiotique relève de la sémantique d'un groupe de textes, de la sémantique d'un texte particulier ou encore de celle d'un usage particulier d'un texte donné. Il permet aussi de délimiter les bornes sémantiques d'une GDT donnée et de faciliter par là son amélioration.

Les langues naturelles sont des objets, qui de façon évidente, résistent fortement aux techniques connues de formalisation.

On doit s'habituer à l'idée d'un système dont les règles changent selon évidemment la portion observée, mais aussi selon l'utilisation qu'on en fait. On peut toujours imaginer une sémantique particulière, fonctionnelle pour un univers langagier donné; plusieurs robots ont démontré qu'il est possible de simuler le fonctionnement d'une langue naturelle pour un univers sémantique restreint. Ces expériences sont intéressantes à titre illustratif, mais elles manquent l'essentiel de ce qui caractérise le comportement normal d'un locuteur: sa capacité de passer d'une sémantique à une autre en adaptant, voire en transformant au besoin les batteries de règles déjà édifiées. Nous devons disposer d'un logiciel qui facilite constamment la construction de nouveaux ensembles de primitifs, de nouveaux axiomes et de nouvelles règles d'inférence, ces éléments étant des variables des procédures programmées et non des constantes.

3. LES DONNEES ET LES ALGORITHMES

Trois objets Déredec retiendront constamment notre attention: les EXFAD, principales structures de rétention des données; les automates, ces machines programmables qui permettent de construire et de modifier les EXFAD et enfin les modèles d'exploration, algorithmes programmés par l'usager pour la fouille et l'extraction de sous-ensembles d'EXFAD. Programmer en Déredec consistera essentiellement à faire interagir ces objets entre eux à l'aide des fonctions Déredec.

3.1 Les EXFAD

Les EXFAD sont des structures arborescentes. Elles se construisent selon un schéma récursif simple représenté dans le tableau ci-bas.

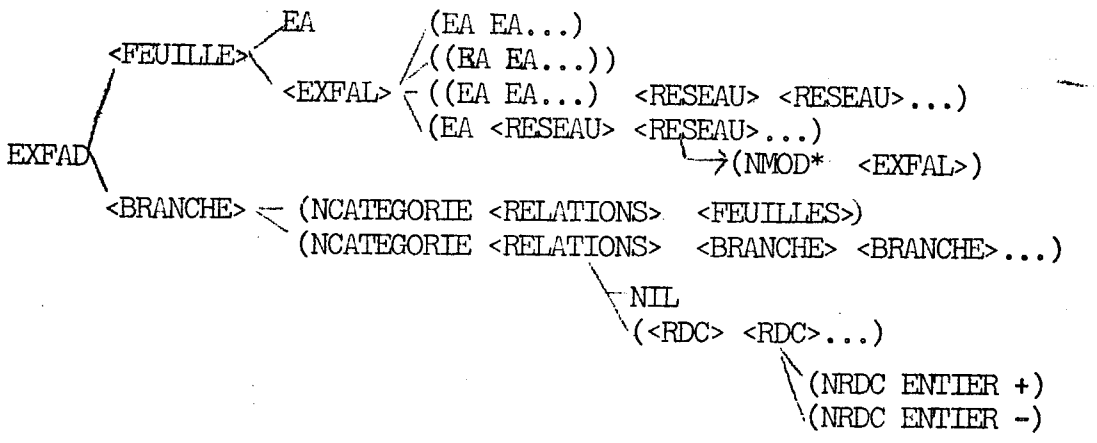
Il faut remarquer que l'usager est rarement appelé à construire lui-même ses EXFAD. Il attellera habituellement des automates Déredec à cette tâche.

Toutefois il doit quand même maîtriser la formule de structuration des EXFAD pour être habilité à bien programmer leur construction par les automates et leur exploration par les modèles.

Dans le tableau, des crochets entourent les concepts descriptifs des structures EXFAD. Certains de ces concepts sont récursifs, cela signifie que l'on devra "passer" plus d'une fois à ces endroits avant d'obtenir l'EXFAD. Dans la structure du tableau, les embranchements signalent des alternatives dans le mécanisme d'écriture.

A l'encontre de ces composants descriptifs, les items du tableau n'ayant pas de crochets peuvent se retrouver présents dans une EXFAD, il s'agit bien sûr des parenthèses ouvrantes et fermantes, des EA (les Expressions Atomiques), des NCATEGORIE (Noms de CATEGORIEs descriptives), des NRDC (Noms des Relations de Dépendance Contextuelle), des NMOD* (Noms de MODèles d'exploration) et des ENTIERS (chiffres).

Lorsqu'un élément est doublé et suivi de trois points, c'est qu'il peut à cet endroit être répété à volonté. Ainsi (EA EA...) signifie l'admissibilité à cet emplacement de (EA) mais aussi de (EA EA EA EA), etc.



On appelle Expression Atomique (EA) le niveau le plus simple d'une EXFAD. Une EA est une EXFAD et une EXFAD contient toujours une EA.

Tous les cheminements dans le tableau se terminent toujours par une EA. Les EA sont des chaînes de caractères mises entre guillemets. Exemple d'une DDT (suite d'EXFAD) à ce niveau primitif de construction:

"Il" "a" "fait" "une" "bonne" "recette" "."

On ne peut imaginer d'EXFAD qui ne soit construite sur une EA. Une EXFAD pourra contenir plus d'une EA. On aura alors affaire soit à une EXFAL (EXpressions de Forme Atomique Liées) soit à une BRANCHE. L'usager utilisera habituellement les EXFAL pour stocker de l'information paradigmatique sur une EA (informations sémantiques ou morphologiques par exemple), alors qu'il utilisera les BRANCHEs pour décrire les relations syntagmatiques (syntaxiques, logiques...) qu'ont entre elles les EA d'une séquence.

Exemples d'EXFAD se ramenant à une feuille:

a) "carte"; pour vérifier le caractère bien formé de cette EXFAD, suivre le cheminement suivant dans le tableau:

<EXFAD> --> <FEUILLE> --> EA

b) ("jambon" "viande" "porc") cheminement:

<EXFAD> --> <FEUILLES --> <EXFAL> --> (EA EA EA)

c)

```
("anticonceptionnel" (RADICAL*  ("conception"))
                     (PREFIXE*  ("anti"))
                     (PLURIEL*  ("s"))
                     (FEMININ*  ("le")))
```

cheminement:

<EXFAD> --> <EXFAL> --> (EA <RESEAU><RESEAU><RESEAU><RESEAU>)

et, pour chacun de ces quatre réseaux:

(NMOD* <EXFAL>) --> (NMOD* (EA))

...par exemple (RADICAL* ("conception"))

d)

```
("Jean" (DEF* ("agent" (TYPE ("humain"))
        (RELATION*
          ("amant"
            (QUI* ("Marie"))
            (OU* ("exemples"
                    (TYPE* (("linguistique"
                             "Chomsky"
                             "générative")))))))
```

Dans une EXFAL, les noms des arcs qui relient entre elles les EA sont des noms de modèles d'exploration (NMOD* dans le tableau), ce sont des identificateurs choisis par l'usager, devant toujours se terminer par un *.

Les modèles d'exploration sont ces algorithmes Déredec programmés par l'usager pour parcourir des EXFAD et en dépister-extraire des sous-EXFAD. Ici, dans le cadre des EXFAL ils ne servent qu'à nommer les arcs des réseaux.

Ce double rôle des modèles d'exploration au sein du Déredec (algorithmes de fouille et appellations des arcs dans les EXFAL) n'est pas une erreur de conception. C'est que d'une première façon, les EXFAL pourront avoir été construites à l'aide de modèles d'exploration. L'utilisation du nom du modèle est alors une trace laissée au moment de cette construction automatique. Que "Jean" soit un "agent" de TYPE* "humain" aurait été ici dépisté par l'exploration antérieure d'une DDT. C'est dans ce sens que l'on conseille d'utiliser les EXFAL pour stocker de l'information paradigmatique d'une EA. Les EXFAL peuvent représenter l'"histoire" du comportement d'une EA dans une DDT. Les EXFAL pourront aussi être construites à la main (en interaction avec le Déredec) et déposées dans des dictionnaires susceptibles d'être projetés sur tout un texte.

L'on verra de plus qu'une EXFAL, grâce au fait qu'elle renferme des appellations de modèles d'exploration, servira elle-même d'algorithme d'exploration au sein de procédures PASC.

Une EXFAL est donc une possibilité d'EXFAD. Elle peut de fait en contenir plus d'une, puisqu'une EXFAL contient des RESEAUX eux-mêmes porteurs d'EXFAL.

Les EXFAL sont des entités Déredec à la fois simples et complexes. Simples par leur formule de construction et complexes par leur double rôle rétentif et procédural au sein du système.

Le niveau le plus simple d'une BRANCHE est celui d'une FEUILLE (EA ou EXFAL) catégorisée, n'ayant pas de RELATIONS (= NIL).

Exemples:

a) (NOM NIL "maison")

cheminement dans le tableau:

<EXFAD> -->
(<BRANCHE> <RELATIONS> <FEUILLE>) -->
(NCATEGORIE NIL EA) -->
(NOM NIL "maison")

b) (ACTION NIL ("manger" (FONCTION* ("survivre"))))

Dans le premier cas la FEUILLE se ramène à une EA et dans le deuxième cas à une EXFAL. NOM et ACTION sont les noms des catégories.

Tout comme c'est le cas pour les NMOD*, on doit distinguer le nom d'une catégorie de la valeur de cette catégorie. On verra plus bas l'utilité de cette distinction.

Deux branches peuvent recevoir une ou plus d'une RELATION si elles sont dominées par une même NCATEGORIE.

Chaque Relation de Dépendance Contextuelle (RDC) contient trois éléments: son nom (NRDC), un nombre entier positif ou négatif qui indique la distance entre les deux BRANCHES reliées par la RDC et enfin un signe + ou - qui indique le sens de la relation. Le + marque la BRANCHE d'où part la relation et le - la BRANCHE qui la reçoit.

Les RDC servent à marquer les relations orientées entre les BRANCHES dans les EXFAD. Exemple d'une EXFAD où trois branches sont dominées par une seule catégorie:

```
(GN NIL (NOM ((RELAT 2 -)) "envie")
        (PREP NIL "de")
        (INFINITIF ((RELAT -2`+)) "partir"))
```

Le nom "envie" reçoit une RDC nommée RELAT de l'INFINITIF "partir". Le nom de la RDC est choisi par l'usager; comme pour les catégories et les modèles d'exploration, on distinguera le nom d'une RDC de sa valeur.

Dans cet exemple, le chiffre ENTIER 2 indique la distance entre les deux BRANCHES. Un ENTIER négatif ordonne un comptage vers le haut (ou la gauche dans l'ordre de la séquence), tandis qu'un ENTIER positif signifie un comptage vers le bas. C'est par ce moyen que l'on pourra dans les explorations retrouver le partenaire d'une RDC. Par ailleurs les signes - et + situés en troisième position de la RDC indiquent respectivement le receveur et l'émetteur de la relation.

Les RDC relient des BRANCHEs de même niveau dans une EXFAD. Une RDC ne peut "traverser" une BRANCHE. Les RDC instruisent des relations privilégiées qu'ont certaines BRANCHEs à l'intérieur d'un noeud commun; la distance séparant ces partenaires (c'est-à-dire le nombre de BRANCHEs déposées entre les deux) étant reléguée à un second plan en

importance. On appréciera à la fois dans la construction et dans l'exploration des DDT, la puissance des moyens combinés RDC et embranchements.

Autres exemples d'EXFAD:

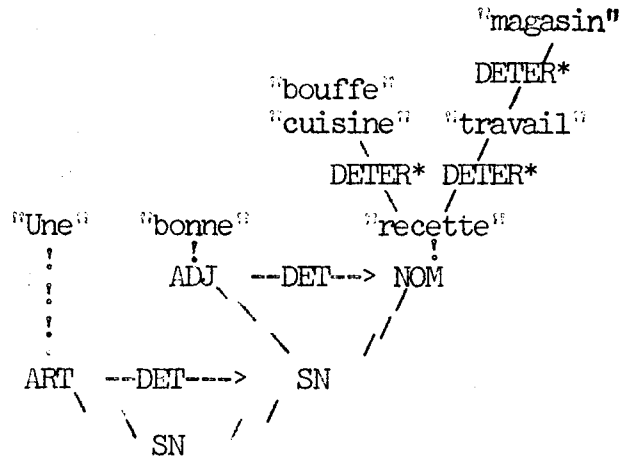
```
a) (SN NIL (SN ((DET 2 -))
               (ART ((DET 1 +)) "Les")
               (NOM ((DET -1 -)) "fondements"))
           (PREP NIL "de")
           (SN ((DET -2 +))
               (ART ((DET 1 +)) "sa")
               (NOM ((DET -1 -)) "métaphysique")))

b) (SN NIL (ART ((DET 1 +)) "Une")
           (SN ((DET -1 -))
               (ADJ ((DET 1 +)) "bonne")
               (NOM ((DET -1 -))
                ("recette" (DETER* (("cuisine" "bouffe")))
                           (DETER* ("travail"
                                       (DETER*
                                          ("magasin")))))))))
```

Dans ces exemples, SN, ART, NOM, ADJ, PREP sont des noms de catégories, et DETER* un nom de modèle d'exploration. DET symbolise une RDC, la relation de détermination.

On appellera couramment "catégorie dominante" d'une EXFAD, la catégorie d'ouverture de l'EXFAD (le SN le plus à gauche dans les deux derniers exemples).

De même, on appellera couramment "expressions atomiques dominantes" d'une EXFAD, toutes les expressions atomiques auxquelles ne sont pas attachées d'EXFAL. De plus, lorsqu'il y a EXFAL, les exepressions atomiques d'ouverture de ces EXFAL, dans l'exemple b), "Une", "bonne", et "recette" sont les expressions atomiques dominantes de cette EXFAD.

Voici la représentation graphique du dernier exemple



Pour que les EXFAD puissent être construites par les automates ou pour qu'elles puissent être fouillées par les modèles d'exploration, les catégories et les RDC devront recevoir une définition ou valeur. Les fonctions actives à l'intérieur des automates et des modèles d'exploration ne manipuleront jamais les noms de ces catégories et RDC mais uniquement leurs valeurs définies.

Les valeurs des catégories et des RDC ont la forme d'une liste ordonnée d'éléments. Ces derniers sont constitués d'un ou de plusieurs caractères. Ainsi la catégorie SN34S2 pourrait avoir différentes valeurs telles: (S N 3 4 S 2), ou (SN 3 4 S2), ou encore (S N 34 S 2), ou même (T RD 23) et ainsi de suite. Tous les tests d'identification d'une catégorie sont exécutés sur ces listes de valeurs. Dès qu'une liste plus petite se réalise dans une liste plus grande (à partir du début de la liste) la comparaison est positive. Ainsi par exemple les catégories suivantes et leurs définitions:

```
SN11 ...  (SN 1 1)
SN121 ... (SN 1 2 1)
SN122 ... (SN 1 2 2)
SN123 ... (SN 1 2 3)
SN2 ...   (SN 2)
SN ...    (SN)
SN1 ...   (SN 1)
SN12 ...  (SN 1 2)
```

Dans ces catégories, SN ramènerait toutes les valeurs, SN12 ramènerait SN12, SN121, SN122 et SN123. SN11 ne ramènera que SN11 ... ainsi de suite.

De plus, une position dans une définition peut être masquée par le caractère _ (le souligné); ainsi SN_2 (défini comme (SN _ 2) ramènera toutes les catégories commençant pas SN et ayant 2 comme troisième élément, quel que soit le contenu du deuxième élément.

Les RDC reçoivent leur valeur d'une façon absolument identique.

Cette façon générale de définir les catégories et les RDC permettra d'une part une très grande économie à la fois dans les batteries de catégories utilisées pour construire une grammaire et dans l'écriture des tests programmés dans les automates ou dans les modèles d'exploration. De plus la définition active d'une catégorie ou d'une RDC étant la dernière fournie (soit au terminal, soit de l'intérieur d'un programme) l'usager peut toujours changer ou faire changer (par ses programmes)

les valeurs de ses catégories ou RDC, ce qui lui procure une très grande flexibilité dans la mise au point de ses algorithmes.

Les NMOD* utilisés dans la construction des EXFAL doivent aussi recevoir une valeur. Ici on distinguera deux types de valeurs possibles, une valeur dite symbolique et une valeur effective. La valeur symbolique est toujours la suivante: (X); les modèles d'exploration à valeur symbolique ne sont utilisés dans les EXFAL qu'à titre de repères "symboliques" pour distinguer les différents arcs reliant les EA entre elles.

Par ailleurs les modèles d'exploration peuvent avoir une valeur effective algorithmique (présentée dans la prochaine section...) et leur insertion dans les EXFAL permettra de transformer alors ces dernières en procédures d'exploration et d'inférence.

3.2 Les modèles d'exploration

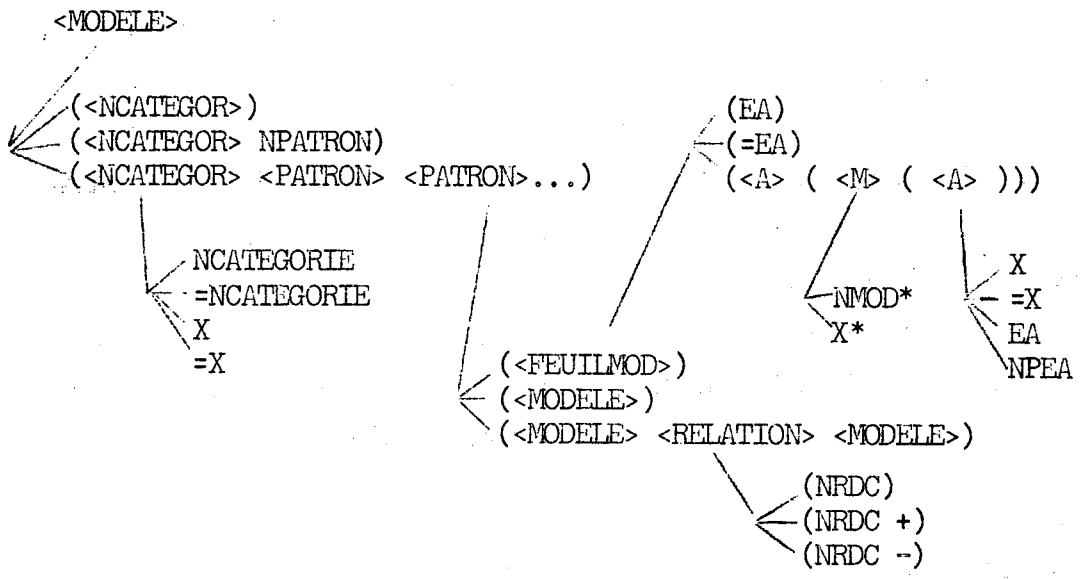
Toutes les EXFAD pourront être fouillées par ce que nous avons appelé les modèles d'exploration.

Ces modèles sont des patrons de fouille et de dépistage (pattern-matching) arbitrairement complexes qui ont pour fonction de rassembler sous des registres ou des fichiers des éléments terminaux ou non terminaux (des sections entières d'un arbre) des EXFAD.

La syntaxe des modèles d'exploration est très souple; elle permet de représenter élégamment les caractéristiques structurales des EXFAD et de dépister de façon très sélective les éléments de ces structures.

Tous les modèles d'exploration ramènent des EXFAD, C'est-à-dire qu'appliqué sur une EXFAD, un modèle ne peut rapporter qu'une sous-section de cette EXFAD qui soit elle-même une EXFAD.

Comme pour le schéma sur la construction des EXFAD, celui sur l'écriture des modèles contient des concepts descripteurs (mis entre crochets) et des éléments terminaux (sans crochets). Seuls ces derniers plus évidemment les parenthèses peuvent se retrouver dans des modèles programmés.



Les items NCATEGORIE (nom d'une catégorie), EA, NRDC (nom d'une RDC), NMOD* (nom d'un modèle d'exploration dans une EXFAL) renvoient directement aux éléments contenus dans une EXFAD. Ils doivent donc dans la construction d'un modèle être remplis par les items équivalents des EXFAD construites par les automates de l'usager.

X dans un modèle masque toute NCATEGORIE (d'une BRANCHE) ou toute EA (d'une FEUILLE). Les FEUILLEs sont spécifiquement explorées par la section FEUILMOD des modèles.

Le signe = lorsque composé avec X, avec une NCATEGORIE, ou avec le signe EA indique la section de l'EXFAD que l'on veut voir rapportée par le modèle d'exploration.

NPATRON (pour Nom d'une suite de PATRONs) et NPEA (pour Nom d'un Patron sur EA) sont des variables qui permettent de "passer" une suite de PATRONS (préalablement construite) au moment de l'évaluation du modèle.

Le contenu des variables occupant les positions NPATRON et NPEA est fourni automatiquement par l'évaluation de la fonction Déredec SOIT. Cette fonction transforme toute suite d'EXFAD en une liste de PATRONS.

Le système permet donc d'inclure à l'intérieur d'un modèle d'exploration des variables dont le contenu est fourni automatiquement au moment de l'évaluation du modèle.

Les RDC marquées dans les EXFAD peuvent être explorées par les modèles. Une RELATION est toujours programmée entre deux MODELES.