Aide à la détection de fautes grammaticales par une analyse progressive des phrases

Christine Fay-Varnier

Institut National Polytechnique de Lorraine

Centre de Recherche en Informatique de Nancy

THÈSE

présentée le 31 janvier 1990

pour l'obtention du titre de

Docteur de l'Institut National Polytechnique de Lorraine
Spécialité Informatique

par
Christine FAY-VARNIER

sujet :

Aide à la détection de fautes grammaticales par une analyse progressive des phrases

Composition du jury :

Président :	Jean-Marie PIERREL
Rapporteurs :	Daniel KAYSER
	Marion CREHANGE
Examinateurs :	Nina CATACH
	Maryse QUÉRÉ
	Daniel COULON

A mes parents

A Denis et Margaux

Remerciements

Je souhaite présenter ici mes remerciements à tous ceux qui m'ont permis d'accomplir ce travail et de le mener à terme :

A Mr Daniel Coulon, professeur à l'Ecole des Mines de Nancy, qui a encadré ce travail et auprès de qui j'ai trouvé conseils, aide et encouragements sans réserve.

A Mr Jean-Marie Pierrel, professeur à l'université de Nancy I, pour avoir bien voulu présider ce jury.

A Mr Daniel Kayser. professeur à l'université de Paris-Nord, qui a accepté d'être rapporteur de cette thèse dans des conditions parfois difficiles et malgré l'importance des nombreuses tâches qu'il assume.

A Mme Marion Créhange, professeur au département informatique de l'IUT de Nancy II, rapporteur de ce travail, à qui je tiens à adresser ma reconnaissance pour la stimulation et les encouragements qu'elle m'a apporté.

A Mme Nina Catach, directeur de recherche CNRS, qui a accepté de bousculer son emploi du temps et de se déplacer à Nancy pour être présente aujourd'hui.

A Mme Maryse Quéré, professeur au département informatique de l'IUT de Nancy II, pour avoir bien voulu participer à ce jury.

Aux membres de l'équipe "Langage Naturel", en particulier, Jean-Marie David, Denis Finck, Laurent Schmitt auprès de qui j'ai toujours trouvé une écoute attentive.

A Dominique Colnet et Samuel Cruzlara mes collègues de bureau pour leur soutien fraternel, leur bonne humeur et leur humour.

Aux membres de l'équipe "Génie Logiciel" qui m'ont offert un sympathique voisinage et ont aimablement toléré que j'utilise leurs machines et partage leurs repas, en particulier Khalid Benali, François Charoy et Laurent Thomas pour leur assistance technique, Henri Basson et Nacer Boudjlida pour leur café, sans oublier Danièle Marchand pour son aide efficace.

A mes autres collègues et amis du CRIN pour leur soutien, en particulier Suzanne Collin, Isabelle Gnaedig-Antoine, Azim Roussanaly.

A tous les membres du département informatique de l'IUT de Nancy II, pour leur accueil et leur sympathie.

Et enfin,

A mes parents qui ont toujours su être disponibles et compréhensifs.

A Denis et Margaux sans le soutien desquels cette thèse n'aurait pu aboutir.

Table des matières

Introduction

L'ordinateur est devenu aujourd'hui un facteur important de notre organisation professionnelle. On y a également de plus en plus recours dans notre vie privée. Le langage codé ne supplante pas pour autant le langage dit naturel, qui demeure la composante majeure au niveau de la communication entre les hommes. L'ordinateur est aussi utilisé pour améliorer la qualité de cette communication, en particulier par l'assistance qu'il peut apporter à la rédaction de documents. Le langage naturel est ensuite un moyen de communication à privilégier entre la machine et les utilisateurs non spécialistes d'une application, notamment pour les systèmes orientés vers le grand public.

Dans ces deux cas de figure, l'analyse d'un énoncé est confronté au problème des *erreurs orthographiques*. D'une part, les concepteurs de traitements de textes souhaitent offrir une maximum d'aide pour la détection des incorrections présentes dans un texte. D'autre part, il est indéniable que la qualité d'un interface homme-machine passe par une certaine convivialité et une souplesse par rapport aux fautes de l'utilisateur, dont les fautes d'orthographe. Enfin, dans les applications à caractère linguistique, un des buts fondamentaux du traitement des données est d'aboutir à une ou plusieurs représentations de leur contenu. Tout phénomène hors des "normes de la langue" connues par le système risque de bloquer cette analyse. Une de ces normes est *l'orthographe* qui régit un certain nombre de conventions pour la combinaison des symboles d'une langue.

Le travail présenté s'inscrit dans le cadre des traitements de textes et porte sur la détection de *fautes orthographiques grammaticales*, plus particulièrement les fautes de fléchissement des mots et certaines fautes d'accord. Les fautes de cette catégorie sont décrites en détail dans le premier chapitre. Ce dernier est consacré à la présentation des différents types d'erreurs rencontrées dans les textes écrits. Nous y décrivons également quelques approches pour les détecter et les corriger de façon automatique. Le lecteur trouvera dans [Veronis 88] une présentation générale des recherches actuellement menées en France par les informaticiens dans le domaine.

En raison du domaine d'application qui a été fixé, les mécanismes à définir pour l'analyse des énoncés se doivent d'être les plus généraux possibles. La langue est considérée comme un système régi par des lois, nous supposons de plus qu'il est possible de décrire séparément :

- son fonctionnement normal (règles du français correct),
- les moyens de détecter son mauvais usage (règles de contrôle).

Les connaissances ainsi traduites en règles sont multiples et variées. Outre la distinction faite classiquement entre les niveaux lexical, syntaxique, sémantique, pragmatique, il faut prendre en considération les différents degrés d'exceptions admis par les règles du français. L'exploitation de ces connaissances dépend également de l'objectif fixé, et notamment du compromis coût (e.g. temps d'exécution) - précision (e.g. détection de toutes les fautes ou des plus grossières) demandé. Ceci justifie que nous ayons adapté à notre problème la notion d'*analyse à profondeur variable* préconisée par Coulon et Kayser [Coulon 80].

Une présentation des notions de représentation progressive des connaissances et de pluralité des stratégies est effectuée dans le début du chapitre 2. Nous montrons ensuite qu'une *analyse progressive* paraît bien adaptée à la détection de fautes au sein d'énoncés en langue française. Cette démarche consiste à démarrer la vérification avec des règles *grossières* pour ensuite *affiner progressivement* le diagnostic obtenu.

Le chapitre 3 est consacré à la représentation des connaissances linguistiques choisie de façon à intégrer les divers aspects d'une analyse à profondeur variable. Nous avons respecté tant que faire se peut la forme progressive des lois exprimées dans les manuels. Les formalismes retenus pour les connaissances de niveaux conceptuels différents : morphologiques, syntaxiques... sont homogènes. Cela offre l'avantage de limiter les mécanismes pour leur exploitation présentés dans le chapitre 4.

Il nous a bien fallu adopter un système de règles pour décrire le français correct, en conformité avec notre modèle de représentation. Nous laissons aux spécialistes de l'orthographe le soin d'en proposer d'autres plus conformes aux théories en vigueur actuellement.

Cette remarque s'applique encore plus en ce qui concerne les *règles de contrôle*. Ces règles permettent de décrire les solutions envisageables pour atteindre un but donné et guider la recherche d'un objectif par la proposition de plans de résolution. En effet, il n'existe pas une unique façon de combiner ces connaissances et de mener l'analyse d'un énoncé. Dans un souci de souplesse et d'adaptabilité, la description des stratégies imaginables est

prévue sous forme déclarative et modulaire. La forme de ces règles appelées *règles de contrôle* est exposée dans le chapitre 4. Nous expliquons également les mécanismes généraux associés permettant la mise en œuvre d'une analyse progressive. Ils prévoient entre autres des étapes de *reprise* après détection d'une situation de blocage.

Le chapitre 5 présente une mise en œuvre des règles de contrôle et les résultats de notre expérimentation. Nous avons montré comment utiliser les outils proposés et combiner les résultats dans des stratégies pour la détection de fautes.

Nous nous sommes attachés à montrer l'intérêt d'une approche progressive et à expliquer la façon de concevoir les règles de contrôle, moins familières pour les non initiés à la programmation.

1

Détection et Correction de fautes d'orthographe

1.1 De l'utilité de corriger les fautes d'othographe

La plupart des auteurs d'ouvrages normatifs et/ou didactiques sur le français *[Grevisse 75] [300 76]* [Catach 82] reconnaissent que l'orthographe, qui en grec, signifie *"bonne façon d'écrire"*, est en grande partie l'effet du hasard et du caprice des hommes, et qu'un certain nombre de ses règles sont arbitraires, gênant ainsi l'acquisition de ses mécanismes.

Mais, d'autre part, le langage, comme l'habillement, comme la manière de vivre, a toujours été considéré comme un signe distinctif du niveau social des hommes. L'Académie Française, instituée par Richelieu, se donne pour tâche de fixer le bon usage qui permet, dit-elle, de "distinguer les gens de lettres d'avec les ignorants et les simples femmes"! Aujourd'hui, l'orthographe a toujours une place prépondérante dans notre vie quotidienne et s'avère source de tracas pour nombre d'entre nous. Elle joue un rôle ségrégatif non négligeable en milieu scolaire, puis professionnel.

Cette dernière constatation justifie le développement d'outils informatiques orthographiques. Les traitements de texte notamment sont sujets à un important développement ces dernières années [Pavard 85]. Ils procurent une aide de plus en plus complète pour la présentation des documents. Cette aide se double de plus en plus par une assistance en ce qui concerne la vérification et/ou la correction des fautes d'orthographe. Même limitée, cette assistance est appréciée des auteurs (écrivains, enseignants, ingénieurs, journalistes, médecins...) qui assurent eux-mêmes la frappe de leurs documents. Elle intéresse aussi beaucoup les éditeurs compte-tenu du coût que représente la relecture d'un document et du coup que portent les coquilles à l'image de marque d'une maison.

Une mauvaise orthographe s'avère également bloquante dans les interfaces homme-

machine, car elle gêne les mécanismes d'appariement utilisés entre autres en interrogation de bases de données. Des erreurs dans l'écriture des noms propres, des termes techniques ou savants posent souvent des difficultés. Mais, comme cela peut nuire à la convivialité du système et paraître inutile de faire remarquer à l'utilisateur qu'il a fait une faute d'orthographe, il semble parfois préférable de la tolérer. C'est le cas par exemple de l'annuaire électronique dans lequel la recherche d'un abonné est approchée eu égard à la prononciation de son nom.

La tolérance aux fautes est aussi un problème fondamental lors de la saisie de textes écrits par lecture optique. En effet cette technique, qui n'a pas encore fini de s'améliorer, n'est pas exempte de défaillances, en particulier la mauvaise reconnaissance de caractères. Il est alors primordial de faire, autant que possible, une correction de ces erreurs [Fisher 76].

Les sections 1.2, 1.3 et 1.4 dressent respectivement un bref inventaire des fautes et des traitements envisagés pour les repérer et les corriger.

1.2 L'orthographe et ses embûches

1.2.1 Introduction

La plupart des ouvrages sur l'histoire et la pédagogie de l'orthographe, distinguent l'*orthographe d'usage* et l'*orthographe d'accord.*

L'orthographe d'usage représente l'ensemble des conventions qui régissent la graphie des mots indépendamment de la fonction qu'ils remplissent dans la phrase, par exemple :

"occurrence" *s'écrit avec 2* "c" *et 2* "r"*;*

Les mots commençant par "ad.." *ont un seul* "d".

L'orthographe d'accord représente l'ensemble des règles qui régissent la graphie des mots selon la fonction qu'ils remplissent dans la phrase, par exemple :

le verbe s'accorde en nombre et en personne avec le nom sujet auquel il se rapporte.

Les problèmes d'accord ne représentent en fait qu'un aspect de tout ce qui concerne les fautes grammaticales.

La typologie utilisée pour la présentation générale des fautes d'orthographe s'appuie

sur cette classification, à laquelle s'ajoute une autre distinction liée, celle-là, à l'étendue de l'analyse nécessaire. L'analyse et les erreurs seront dites :

- **lexicales** lorsque l'examen d'un mot à lui seul permet de déceler la présence et la cause d'une faute ;

- **grammaticales** lorsque l'erreur n'est éclaircie qu'après avoir pris en considération une portion de l'énoncé correspondant à une structure grammaticale.

Enfin une troisième distinction est également souvent mentionnée [Chomsky 65] concernant :

- la **compétence** : connaissance des règles normatives ;

- la **performance** : l'emploi effectif de la langue dans des situations concrètes.

1.2.2 Les fautes lexicales

Ce paragraphe est consacré à la présentation des fautes lexicales. Un mot qui contient une faute lexicale est une chaîne de caractères dont la graphie ne respecte pas les conventions de l'orthographe d'usage. Elles ont des causes variées. Les *fautes d'usage* sont généralement dues à une mauvaise connaissance de l'orthographe d'usage (fautes de compétence). Les *fautes typographiques* sont des fautes de performance qui affectent la graphie d'un mot.

Les fautes d'usage sont nombreuses et variées :

Qui n'a jamais fait d'erreur sur l'écriture de *parallèle* ou de *occurrence* ?

Qui peut expliquer pourquoi *connexion* s'écrit avec un *-x* et non pas avec *-ct* ?

Pourquoi *chariot* n'a-t-il qu'un *r* tandis que *charrette* en a deux ?

Leur origine est diverse.

- Si l'étymologie et l'histoire [Catach 82] peuvent parfois permettre de choisir, la bonne orthographe, comme pour *dégoûter* (du latin *gustus*) et *dégoutter* (du latin *gutta*), certains usages de la langue française ne peuvent pas toujours s'expliquer. L'inexistence de règles, ou leur complexité, explique que l'on ait recours à la comparaison avec les éléments d'un lexique pour détecter les fautes d'usage. [Pérennou 86] indique le principe et les modes de recherche et de comparaison actuellement mis en œuvre.

Une autre difficulté de l'orthographe d'usage provient du manque de concordance entre l'oral et l'écrit, entre phonème et graphème en particulier. Le français fourmille d'homophones du type *sain, sein, saint, ceint* où le son $\tilde{\varepsilon}$ se retrouve sous des formes différentes à chaque fois *[Thimonnier 79].* Une grande partie des fautes d'usage résulte donc d'une mauvaise transcription de l'oral, ce qui justifie l'usage de transcriptions pseudo-phonétiques.

exemples :

♠ *Pourquoi ai-je fait celà(cela) ?*[1]

♠ *Cela permet [...] de pouvoir déclancher(en) une action sur une seule information [..].*

- L'utilisation d'un clavier pour la saisie d'un énoncé provoque un certain nombre de fautes de frappe appelées *fautes typographiques.* Ces fautes sont fortement liées à la disposition des lettres sur le clavier et ne proviennent pas de lacunes de l'utilisateur sur l'orthographe.

exemple :

♠ *L'ensemble de l'édifice, y compris les duex(eu) tours, ne fut achevé qu'en 1693.*

Ces fautes ont été beaucoup étudiées car elles représentent 80% des erreurs dans les textes [Durham 83]. En général, les auteurs [Damereau 62] [Muth 77] [Peterson 80] classent les erreurs de la façon suivante ;

- *effacement* d'une lettre,
- *insertion* d'une lettre,
- *substitution* d'une lettre,
- *permutation (interversion)* de deux ou plusieurs lettres.

1. Les fautes sont présentées dans les exemples de façon à faire apparaître l'endroit où se situe l'erreur (soulignement) et sa correction (entre parenthèses).

[Pérennou 86] prend également en compte :

- la *coupure* d'un mot (insertion d'un blanc),
- la *soudure* de deux mots (effacement d'un blanc).

Ces anomalies représentent en effet dans son corpus un pourcentage de 20% des fautes.

La table 1.1 page 14, largement inspirée de [Pérennou 86], présente les différentes sortes de fautes lexicales. La première colonne *(typographie)* indique les fautes dont l'origine peut être attribuée à une erreur de frappe, la deuxième regroupe des fautes qu'on peut imputer soit à une méconnaissance de l'orthographe, soit plus prosaïquement à une erreur de frappe. La troisième élargit les exemples à des fautes non lexicales, dans lesquels la présence d'une faute n'est pas certaine et serait due à une confusion au niveau de la compréhension de l'énoncé (ou tout simplement encore à une faute de frappe).

Les erreurs résultant d'une lecture optique sont de la même façon indépendantes de la connaissance orthographique de l'auteur. Il doit être également possible de les classer ici en tenant compte des aléas du procédé d'encodage.

1.2.3 Les fautes grammaticales

Les *fautes grammaticales* représentent les fautes qui résultent de la mauvaise utilisation ou de l'omission d'une règle donnée dans la grammaire de la langue. Cette dernière intègre plusieurs niveaux de connaissances qui donnent naissance à plusieurs types de fautes grammaticales.

La morphologie

La morphologie est l'étude des diverses formes que peut prendre un mot pour traduire notamment les accords, et les règles de morphologie régissent le fléchissement des mots pour marquer le pluriel, le féminin, les conjugaisons...

exemple :

∇ *Les noms en* "-au" *forment leur pluriel en ajoutant un* "-x".
Exceptions : landau *et* sarrau. *[Larousse 82]*

\ cause type \	typographique	typo ou orthographique	ambiguïté d'interprétation
effacement	difficle(cile)	brilance(ll)	27 candidats en cours... (en course)
insertion	hiérarichie() éémoi()	cabanne() bazard()	ces cachots épuisants() la ballade de Chri...()
substitution	reduge(f) luvret(i)	collire(yr/in) précédante(en)	McEnroe a publié(ou) ses déboires
permutation	Firtz(ri) territorail(ia)	dalhia(hl) infractus(ar)	mine de rein(ien) sauce aux carpes(pr)
coupure	bea_ucoup	anti_rouille	Duhamel fut un animiste(una) en_fin ()
soudure	unami(un a)	Lepage(Le page)	Il enfume(en f)

Tableau 1.1. *Exemples de fautes classés.*

exemple de faute de flexion :

♠ *Je voulais rester au lit pour écouter les chants des oiseaus(aux)".*

La Syntaxe

L'étude de l'agencement des mots dans une phrase en groupes de mots (ou syntagmes), et de leurs fonctions respectives, constitue la **syntaxe.**

exemple :

∇ *Le groupe du nom simple est formé d'un déterminant suivi d'un nom [Dubois 73].*

Les fautes qui découlent du non respect de ces règles sont des fautes de construction appelées par extension *fautes syntaxiques.*

exemple :

♠ *Elle se se() souvient être arrivée en gare de Nancy.*

La syntaxe définit également les variations des mots selon leurs fonctions dans l'énoncé. La variation d'un constituant de la phrase peut en effet entraîner la variation d'un de ses autres constituants, c'est ce qu'on appelle un **accord**. Il porte la plupart du temps sur le genre, le nombre, la personne des constituants concernés.

exemple :

∇ *Dans une proposition, le sujet s'accorde avec le verbe auquel il se rapporte [Bescherelle 84].*

exemple de fautes d'accord :

♠ *Ces dernières représente(ent) ce que l'on appelle l'ensemble des dépendances de l'équation sémantique.*

Les fautes grammaticales mettent en jeu un grand nombre de règles, dont beaucoup sont locales [Fouqueré 88]. Ceci explique que les produits commerciaux n'apportent qu'une aide partielle à ce niveau, même les plus récents comme par exemple le logiciel québécois Hugo.

La sémantique

Il existe des fautes difficiles à déceler car elles modifient le sens d'un énoncé, et ne peuvent être détectées sans comprendre au moins partiellement le texte.

exemple :

♠ *[...] autre avantage d'une base lunaire : étant donné que la gravité est six fois moins importante _(que) sur terre, l'envol des fusées...*

On pourrait compléter cette énumération en parlant du "discours". Il existe en effet des règles de la narration, du récit, de la description, de la démonstration ... Ne pouvant

déjà pas déceler en général les fautes sémantiques, nous ne rechercherons pas celles relevant de la **rhétorique** !

Nous avons classé les fautes d'après :

- les traitements qui permettaient de les détecter ;

- les causes qui pouvaient expliquer leur présence.

Nous allons procéder de la même façon pour indiquer les méthodes implémentées. Le lecteur trouvera une présentation plus détaillée de ces méthodes dans [Morgan 70] [Peterson 80] [Pérennou 86] [Veronis 88]. Nous nous contenterons ici d'en donner un aperçu simplifié.

1.3 La détection des fautes sur les mots considérés isolément, et leur correction

1.3.1 Détection

Les sytèmes qui traitent les mots isolément de tout contexte sont les plus nombreux actuellement. Les études sur ce sujet sont en effet assez anciennes [Damereau 62] et ont été nombreuses. On distingue deux grandes catégories de solutions :

- les unes partent d'une description des fautes possibles,

- les autres partent d'une description des formes correctes des mots de la langue.

Ces deux approches ne sont pas forcément exclusives.

Quelle que soit la démarche suivie, deux voies sont encore possibles pour décrire les connaissances utiles :

- soit effectuer un recensement préalable de toutes les formes (fausses ou correctes selon la solution choisie) ;

- soit employer des règles générales accompagnées le cas échéant de règles d'exception.

Le recensement des formes ou la définition des règles peut résulter d'études statistiques et/ou linguistiques.

a) Les méthodes statistiques

Les méthodes présentées dans ce paragraphe s'appuient sur des informations statistiques recueillies soit au niveau du texte analysé, soit au niveau plus général de la langue. Elles n'utilisent pas du tout de lexique.

• Fréquences de mot dans un texte

Une première génération de vérificateurs partaient du principe que dans un texte, les mots qui apparaîtront le moins sont ceux qui sont d'usage rare et dont l'écriture est moins familière, ou bien ceux sur lesquels il y a une faute. La démarche suivie consiste à dresser une liste exhaustive de tous les mots distincts reconnus dans le texte. Cette liste est ensuite triée par ordre croissant de fréquences d'apparition, c'est à dire les mots les plus suspects en premier, puis présentée à l'utilisateur pour qu'il certifie leur bonne écriture [Damereau 62]. Une amélioration de ce mécanisme consiste à ne pas présenter les mots dont la fréquence d'apparition est au-dessus d'un certain seuil.

Malheureusement, les fautes dues à l'ignorance d'un mot et qui se retrouvent pour toutes les occurrences de ce mot sont difficilement détectées. D'autres méthodes ont été proposées pour pallier cet inconvénient.

• Les n-grammes

Pour repérer les formes les plus particulières d'un texte, d'autres systèmes utilisent la technique des n-grammes. Un *n-gramme* est une suite de n lettres dans une chaîne de caractères. Par exemple, *lettre* donne : 5 digrammes : *le, et, tt, tr, re,* et 4 trigrammes : *let, ett, ttr, tre.*

[Peterson 80] a calculé qu'en anglais, en comptant le blanc d'apostrophes, on a 28 lettres qui donnent : 784 digrammes, 21952 trigrammes, mais que seulement une portion d'entre eux sont valables dans la langue. Dans un texte courant en anglais, on trouve environ 550 digrammes différents c'est à dire 70% du nombre de digrammes possibles et environ 5000 trigrammes différents, soit environ 25% du nombre de trigrammes possibles.

Un certain nombre de programmes, dont TYPO [Morris 75], ont exploité ces remarques pour étendre les solutions présentées dans le paragraphe précédent [Riseman 74] [Cornew 68]. TYPO calcule les fréquences d'apparition des digrammes et trigrammes pour les mots d'un texte. Un indice est calculé avec ces fréquences pour chaque mot différent de

la liste créée par TYPO. Le but de cet indice est d'attribuer à un mot une valeur d'autant plus grande qu'il contient des digrammes et des trigrammes rares.

Cette technique peut être améliorée par l'adjonction d'une liste de mots corrects et fréquemment utilisés, la liste soumise à l'utilisateur étant réduite d'autant.

Les observations statistiques peuvent intervenir aussi dans la conception d'un lexique, comme nous allons le présenter avec les méthodes qui s'y réfèrent.

b) Les méthodes lexicales

• Stockage des formes correctes

L'accès à un grand nombre d'informations stockées s'est nettement amélioré ces dernières années. Cela a permis le développement de systèmes de vérification automatique de textes basés sur un lexique. Celui-ci, indépendant du texte, conserve tous les mots de la langue susceptibles d'être employés. La démarche suivie par ces vérificateurs consiste à comparer chaque mot du texte à la liste des formes correctes recensées et contenues dans le lexique.

C'est ainsi qu'opèrent les sytèmes disponibles actuellement sur le marché, parmi lesquels : Writing-Assistant (IBM-PC), Word3 (Micro-soft), Wordstar2000 (Micro-pro), Personnal-Writer, Visio3-PC (IBM), Orthogiciel (Larousse pour MacIntosh), Spell sur Unix.

L'élément clé pour la performance de tels systèmes est dans ce cas le dictionnaire. Des études ont été menées afin de l'organiser le mieux possible. Elles ont permis d'obtenir la table ci-dessous qui indique le recouvrement de la langue anglaise par un vocabulaire restreint des mots les plus couramment utilisés [Peterson 80].

200 mots les plus fréquents	couvrant :	50% des textes
2000 mots spécifiques	couvrant :	45% des textes
20 000 mots du vocabulaire général	couvrant :	5% des textes

Tableau 1.2.

Ainsi les 256 mots les plus courants de la langue anglaise permettent de reconnaître plus de la moitié des occurrences des mots d'un texte.

De façon similaire, [Catach 84b] a proposé une liste de 1600 mots français et de leurs formes fléchies (4000 formes en tout) les plus utilisés. Ils couvrent en moyenne 90% des potentialités d'occurrences dans n'importe quel texte en langue française.

Peterson a envisagé un dictionnaire à plusieurs vitesses. Il prévoit un lexique réduit des 258 mots les plus fréquents de l'anglais et un lexique des termes spécifiques du sujet ou du document traité, tous les deux d'accès rapide. Le reste du dictionnaire, de volume important mais avec une faible fréquence de consultation, a un support d'accès moins rapide.

• Utilisation de règles morphologiques

Certains programmes proposent également une réduction de la taille du lexique grâce à l'utilisation d'un ensemble de règles sur la composition morphologique des mots (affixes-suffixes). On rencontre cette technique notamment dans les systèmes de DEC (SPELL) et de [Pérennou 86].

1.3.2 Correction des fautes sur le mot isolé

Il s'agit maintenant, non seulement de signaler à l'utilisateur des mots qui posent problème, mais de lui proposer une ou plusieurs solutions de rechange en recherchant des formes voisines selon un critère donné [Pérennou 86].

Certains mécanismes de correction s'inspirent des causes possibles de l'erreur tandis que d'autres exploitent plutôt ses conséquences.

• Construire la correction

Une des techniques de correction automatique consiste à s'appuyer sur les causes possibles de la faute pour construire des formes proches du mot erroné. Dans le cas où on choisit de ne corriger qu'une seule faute par mot, une procédure consiste à engendrer l'ensemble des variantes du mot erroné par une transformation : inversion, suppression, insertion... L'intersection entre cet ensemble et le lexique est proposée à l'utilisateur comme correction possible [Damereau 62][Morgan 70].

L'algorithme de [Viterbi 67] utilise des statistiques sur les n-grammes et construit un modèle de type markovien pour introduire des substitutions dans les textes, dans le but de retrouver les formes correctes des mots le composant.

• Recherche d'invariants

Une autre façon de procéder consiste à rechercher dans le lexique un ensemble de chaînes peu différentes du mot erroné. La recherche dans le lexique est une recherche approximative. Pour cela on met en œuvre des techniques permettant de transcrire le mot correct et ses variantes erronées en une même représentation canonique ou clé. Un lexique des formes canoniques permet ainsi de proposer les différentes corrections envisageables pour un mot.

La clé peut être :

- un *squelette* du mot, supposé résistant aux fautes, composé de certaines de ses lettres choisies pour leur pouvoir discriminant. Cette technique est très ancienne [Blair 60] [Davidson 62]. Depuis, différents arrangements de squelettes ont été proposés [Veronis 88] ;

- une *transcription phonétique* du mot, sensée elle aussi être résistante aux erreurs dues aux diverses façons de transcrire des prononciations semblables.Cette méthode est particulièrement appropriée pour le traitement des noms propres. Les règles pour interpréter les requêtes dans l'annuaire électronique tiennent compte, entre autres, de cet aspect.

Pour établir un ordre dans la construction des variantes, ou la restitution des mots associés à une forme canonique, il est possible d'avoir recours à une métrique [Lenvenshtein 66]. Une mesure de vraisemblance entre mots est définie à partir des transformations élémentaires (substitution, insertion, suppression et inversion de caractères) nécessaires pour passer du mot erroné au mot du lexique. A chacune de ces transformations est associé un poids qui peut être constant, dépendant des caractères mis en jeu, ou encore probabiliste.

• Organisation du lexique

D'autres réalisations, comme [Hall 80], proposent de structurer le lexique de façon à faciliter et/ou accélérer les recherches, en optimisant le nombre d'accès. [Morgan 70] propose par exemple le regroupement des mots contenant le même nombre de lettres. [Muth 77] celui des mots de même initiale tandis que [Morgan 70] encore construit ses partitions à partir des deux premières lettres des mots. D'autres combinent les critères de longueur, de valeur des premières lettres ou des trigrammes apparaissant dans le mot [Fournier 84]. La recherche d'un mot est orientée dans les sous-lexiques opportuns.

Les résolutions proposées pour définir les partitions ne donnent pas toujours une répartition équilibrée des mots. De plus, ces solutions ne sont pas tolérantes envers un certain

nombre de fautes. Plutôt que d'être employées seules, ces méthodes gagnent à compléter les solutions présentées précédemment.

• Combinaison de différentes approches

Il n'existe pas en fait de solution unique. Il est alors intéressant de combiner ces différentes approches, comme dans le système Vortex de Perennou [Pérennou 86].

La particularité de ce système est qu'il utilise trois canaux distincts qui modélisent les fautes orthographiques, typographiques et les fautes de transposition. Ces modèles sont fondés essentiellement sur une approche probabiliste. Un mot erroné est traité à l'aide de combinaisons produites successivement par chacun des trois canaux. Le lexique utilisé est représenté au moyen d'un alphabet particulier dont les éléments sont les groupes de lettres posant des problèmes orthographiques.

exemple :

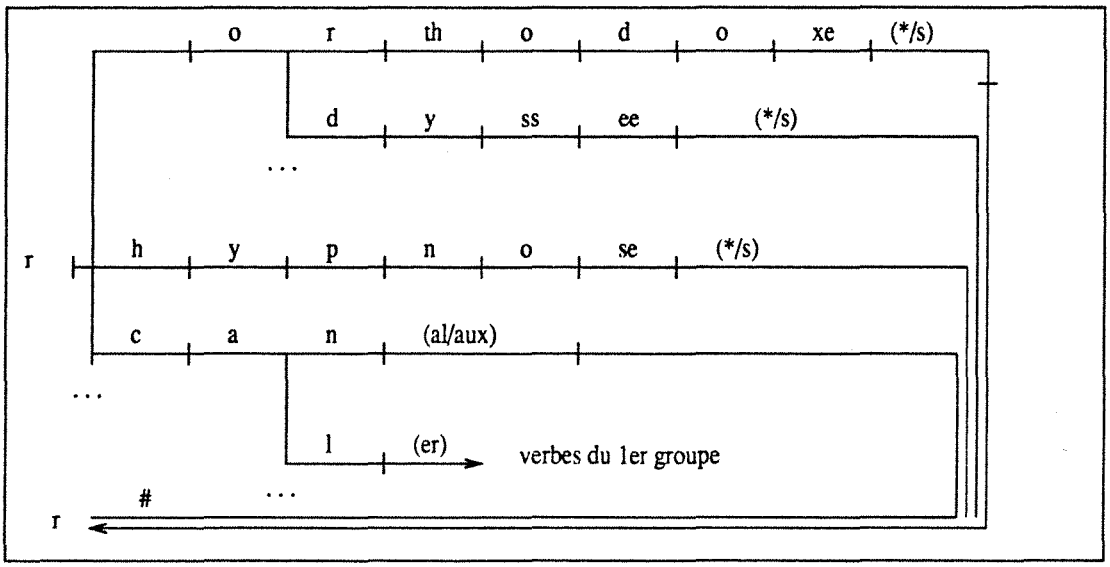


Figure 1.1. Exemple d'arbre lexical dans le système Vortex

On remarquera que le lexique a été pré-traité par un composant morphologique.

• La correction syntaxique

Dans toutes les techniques présentées précédemment, les informations utilisées sont extraites au niveau du mot : proximité, morphologie... Il est certain pourtant que les

niveaux considérés comme "supérieur" (syntaxique, sémantique, pragmatique) peuvent apporter des indications pertinentes sur la correction à effectuer.

Prenons l'exemple suivant :

"L'enfant man ge de la bouillie."

deux chaînes ne sont pas reconnues : *" man "* et *" ge "*. Il est intéressant de remarquer qu'il manque un verbe entre le groupe nominal *"l'enfant"* et le groupe prépositionnel *"de la bouillie"* pour former une phrase correcte pour la grammaire du français. La correction peut être orientée vers la concaténation des deux chaînes inconnues pour n'en former qu'une. On vérifiera ensuite que cette nouvelle chaîne est un verbe de la langue.

En utilisant les techniques des réseaux syntaxiques, [Fouqueré 86] a développé une telle méthode. Elle consiste à observer les suites de transitions possibles pour aboutir d'une partie reconnue de la phrase à une autre et elle complète ainsi une correction orthographique basée sur la recherche de voisins. Cette technique s'avère intéressante pour la correction des fautes de coupure ou de soudure de mots.

Même si, comme on l'a ressenti à travers ce dernier propos, ce découpage (lexical, morphologique, syntaxique, sémantique...) est quelque peu arbitraire, nous débutons une nouvelle section pour présenter les problèmes dits de niveau "supérieur".

1.4 La détection et la correction des fautes en contexte

La détection des fautes grammaticales présentées dans le §1.2.3 nécessite l'étude de la structure de la phrase par rapport à un modèle donné de la langue. Dans certains systèmes, la correction des fautes est proposée dans une étape ultérieure spécifique, après rejet de l'énoncé par l'analyseur [Granger 84] [Heidorn 82]. La plupart des auteurs reconnaissent cependant l'intérêt de continuer l'analyse après échec [Hayes 80] et de tenter de préciser tant que faire se peut le contexte dans lequel se situe l'erreur. Dans ce cas, la correction est intégrée à l'analyse proprement dite afin de pouvoir exploiter toute information utile sur les accords, sur les contraintes syntaxiques, sémantiques ...

Pour la langue française, à notre connaissance les fautes d'accord sont les fautes grammaticales dont le traitement informatique a été le plus étudié. [Fouqueré 88] propose cependant un analyseur tolérant du langage naturel testé sur des énoncés en français et ne

se limitant pas aux erreurs d'accord. Dans les applications sur la langue anglaise qui prennent souvent en compte les problèmes de constructions syntaxiques erronées, la sémantique prend en contrepartie une part importante au moment de la correction.

1.4.1 Détection des fautes d'accord

Dans les sytèmes présentés ci-dessous, les mécanismes de détection et de correction sont fortement liés, et nous n'en ferons pas d'exposés distincts. Les méthodes employées diffèrent en grande partie par les techniques d'analyse syntaxique qu'elles mettent en œuvre.

[Granger 84] propose une analyse réduite à celle de syntagmes locaux : groupes nominaux, constructions de verbes..., [Richard 86] propose une analyse complète des phrases axée sur la vérification des accords des participes passés. Ces deux systèmes sont écrits en PROLOG, et utilisent le formalisme des "Grammaires à Clauses Définies" (DCG) [Pereira 80].

Les règles qui décrivent les agencements autorisés des mots dans la phrase sont complétées par les contraintes d'accord qui leur sont attachées. Des informations sémantiques peuvent également apparaître sous forme de traits sémantiques, attachés aux constituants de la phrase. Ces traits permettent de lever des ambiguïtés apparues au cours de l'analyse ou bien s'avèrent indispensables pour vérifier certaines conditions.

Ces deux réalisations adoptent une stratégie générale en vue d'optimiser la recherche des fautes. Le système de [Granger 84] procède en plusieurs étapes. Une analyse syntaxique pour déterminer la structure de la phrase est tout d'abord effectuée. En cas d'échec, on suppose que ce dernier peut être imputable à une faute d'accord, l'analyse est alors réactivée en inhibant dynamiquement les contraintes d'accord en genre et en nombre. Si l'analyse aboutit, c'est qu'il y a effectivement présence d'une faute d'accord. Alors seulement un système de réécriture est déclenché sur les mots de la phrase, pour corriger les accords défectueux.

Le système de [Richard 86] corrige les fautes au fur et à mesure que s'effectue l'analyse syntaxique. Pour cela il fixe a priori la propagation des attributs grammaticaux, quitte à ne pas respecter toutes les possibilités offertes par la langue.

exemple

∇ *Dans un groupe nominal (gn) constitué d'un déterminant suivi d'un nom, le genre et le nombre du gn sont hérités de ceux du déterminant.*

Si le genre du déterminant n'est pas connu, l'héritage se fait à partir du genre du nom. Si celui du nom n'est pas connu non plus, le masculin est retenu par défaut.

1.4.2 Correction syntaxico-sémantique

Les systèmes présentés dans cette section sont plus généraux, dans le sens où ils ne se limitent pas en règle générale à proposer des corrections au niveau syntaxique. Comme nous l'avons déjà dit ci-dessus, les systèmes sur la langue anglaise s'appuient pour cela sur une sémantique connue du domaine de travail. [Fouqueré 88] présente deux méthodes qui, au contraire, sont totalement guidées par la syntaxe.

• **Correction syntaxique**

Le premier système présenté dans [Fouqueré 86] et [Fouqueré 88], déjà introduit dans le §1.3.2, est basé sur un mécanisme d'analyse de type *Augmented Transition Networks (ATN)* [Woods 70]. L'objectif recherché est de pouvoir effectuer une analyse non linéaire de la phrase (c'est à dire qui ne découle pas uniquement d'une lecture gauche-droite), afin d'être mieux armé contre les problèmes liés à de mauvaises constructions syntaxiques. La démarche suivie consiste à rechercher des parties sûres d'analyse de façon à s'en servir comme îlots de confiance à partir desquels l'analyse complète pourra s'effectuer.

Le système se compose d'un contrôleur gérant le développement de modules effectuant des analyses partielles. Le travail sur chaque analyse partielle s'effectue en deux temps :

- recherche des hypothèses permettant la pousuite de l'analyse ;
- application de l'hypothèse de moindre coût.

L'analyse développée revient à choisir à chaque étape un chemin à suivre (ce qui est fait de façon non déterministe) et soit à poursuivre l'analyse lorsque c'est possible, soit à utiliser une technique de déblocage lexicale ou syntaxique. L'application d'une technique de déblocage lexicale consiste à rechercher dans le lexique les voisins d'un mot donné, de concaténer des chaînes successives de la phrase à analyser ou de décomposer une

chaîne. Avec une technique de déblocage syntaxique, il s'agit de trouver dans le réseau de transitions de l'automate un chemin minimal aboutissant à une autre analyse partielle en cours ou à une catégorie exploitable. Les modules d'analyse partielle admettent comme paramètres le sens de l'analyse (droite-gauche ou gauche-droite).

Ce type de démarche permet de corriger les diverses phrases contenant une ou deux fautes orthographiques ou syntaxiques, soumises au cours de l'expérimentation. Elle a l'inconvénient d'après l'auteur lui-même d'émettre trop d'hypothèses tant au niveau lexical que syntaxique.

• Correction orientée-acteur

Le second système présenté également dans [Fouqueré 87] s'appuie sur un modèle grammatical particulier, la Grammaire A Configuration Minimale (GACM), adapté au problème de la correction orthographique et fondé sur une description par défaut des symboles grammaticaux.

La correction recherchée en cas d'erreur est celle qui minimise le coût des transformations qu'il faut accepter pour rendre compatible l'énoncé avec une phrase de la langue. La démarche appliquée consiste à effectuer une analyse ascendante du texte sans ordre spécifique pour l'analyse des groupes correspondant à une structure d'un niveau donné, et à la suspendre dès qu'il y a ambiguïté grammaticale.

Le correcteur cherche alors, comme dans la démarche précédente, à exploiter les résultats partiels et les définitions portant sur les constituants (ou entités). Les actions correctives sont guidées par une génération descendante de contraintes fournies par la grammaire du français. Dans ce contexte, les structures d'exceptions ne sont envisagées que dans une analyse montante.

exemple :

Une phrase est, a priori, constituée d'un Groupe Sujet et d'un Groupe Verbal ; s'il existe un Complément alors on envisagera une autre structure pouvant convenir.

Il s'agit en effet, d'obtenir l'ensemble des contraintes par défaut, c'est à dire la description a priori la plus importante pour le type de phrases considérées.

Le correcteur a été développé à l'aide des concepts d'acteurs, chaque entité (mot, concept grammatical ou sémantique) intervient dans une analyse comme acteur de la représentation à créer du texte soumis. Il est alors possible d'associer à ces entités des méthodes de contrôle de l'analyse et de lancement d'hypothèses de correction. Le contrôle de l'analyse est donc local, le contrôle global existe cependant, mais est limité à l'essentiel : gestion de la file d'attente des actions (messages à lire ou méthodes à exécuter) et répartition des temps d'exécution des actions locales.

Dans ce sytème, on distingue deux types de correction : une correction contextuelle qui est associée à l'entité et aux erreurs qui en dépendent ; une correction standard qui peut intervenir sur n'importe quelle entité, c'est par exemple la correction sur les fautes d'usage.

- **Correction syntaxico-sémantique**

Les approches présentées ici sont toutes basées sur l'utilisation de contraintes sémantiques pour tenter de proposer une correction cohérente avec le domaine considéré.

Une première façon de procéder est de prévoir une liste d'actions heuristiques à entreprendre en cas de blocage de l'analyse. C'est ce qui est effectivement mis en œuvre par [Weischedel 83] qui utilise des ATN. A chaque état d'un automate supposé déterministe, est associé le comportement à suivre en cas d'échec, comportement défini à l'aide des contraintes sémantiques de l'automate. Dans [Hayes 84] et [Minton 85], l'analyse effectuée est sémantique. Une liste d'actions similaire est associée à chaque règle sémantique. Le système de contrôle privilégie l'action de plus faible coût de façon à obtenir la solution sans faute s'il en existe une.

Une autre approche exploite la notion d'appariement ("pattern-matching"). Mise en œuvre par [Hayes 80], elle consiste à tenter de compléter les structures partielles obtenues après échec de l'analyse par les termes voisins du texte en effectuant un appariement des composants.

L'inconvénient de ces méthodes réside dans la nécessité de pouvoir donner une représentation sémantique du domaine de travail. Outre la difficulté d'obtenir une telle représentation sur un domaine donné, cela limite leur utilisation à des domaines sémantiquement restreints.

1.4.3 Vers l'assistance à la rédaction

L'assistance à la rédaction dont nous avons évoqué l'intérêt, ne doit pas se limiter à une vérification orthographique des textes. Un système d'IBM, nommé Critique (anciennement Epistle) [Heidorn 82] analyse les phrases en vue d'apporter des remarques sur la rédaction et le style. Son domaine d'application est principalement le courrier d'affaire, et il se situe dans un cadre de bureautique.

Un analyseur de langue naturelle très performant constitue le principal élément de ce système. Il s'appuie sur une grammaire hors-contexte de la langue dont les règles sont *augmentées* de conditions supplémentaires (Augmented Phrase Structure Grammar). L'analyse effectuée est une analyse depuis les données vers les buts.

La détection des fautes grammaticales s'effectue en 3 phases :

- analyse syntaxique de l'énoncé en utilisant les règles dans leur intégralité;

- dans le cas où la phrase n'a pas été reconnue, nouvel essai de filtrage en relâchant la vérification de certaines contraintes;

- en cas de succès de l'étape précédente, prendre note des conditions dont la vérification a été négligée et envoyer des messages à l'utilisateur par l'intermédiaire de l'interface associée à un éditeur de texte.

Les fautes signalées sont principalement des fautes d'usage, des erreurs d'accord nom-déterminant, sujet-verbe, certaines concordances de temps des verbes.

Dans une seconde phase, Epistle-Critique produit une critique générale du style du texte : tournures maladroites, expressions à éviter, répétitions... et fournit un indice de lisibilité ainsi que quelques informations statistiques (pourcentage passif-actif, termes souvent utilisés..). Les diagnostics d'erreurs de style sont obtenus à l'aide d'une analyse descendante sur un autre ensemble de règles.

* * *

Bien qu'intéressés par une large couverture de l'assistance apportée à l'utilisateur, nous avons limité notre projet à la détection de faute de nature grammaticale, résultant d'une méconnaissance ou d'un mauvais emploi de règles morphologiques et d'accord, comme nous le présentons dans le chapitre suivant. Notre but principal était d'expérimenter une stratégie de détection (chapitre 5) adaptée à une description progressive des connaissances (chapitre 3).

2
Présentation du projet

L'étude présentée comporte deux aspects. Nous souhaitons d'un point de vue pratique, réaliser un système de détection et d'aide à la correction de fautes d'orthographe. Un objectif plus théorique est d'étudier l'intérêt et la mise en œuvre de raisonnements à profondeur variable dans une telle application. Le début de ce chapitre consiste à définir plus précisément la tâche fixée. Avant de souligner dans la 3ème partie l'intérêt que semble apporter la notion de profondeur variable à la détection de fautes d'orthographe, nous présentons ce que recouvre de façon générale cette notion.

2.1 La tâche projetée

2.1.1 Détection de fautes

Nous nous situons dans le cadre de la saisie de textes à l'aide d'un clavier, pour l'utilisation par exemple d'un traitement de textes. Il s'agit de repérer les fautes en vue d'apporter une aide à la rédaction.

De nombreuses solutions satisfaisantes ayant été apportées pour le traitement des fautes sur le mot isolé de tout contexte grammatical (§1.3), nous avons supposé résolu ce problème. Nous avons porté notre effort sur les erreurs nécessitant une vérification du bon usage des lois de construction de la langue.

- les *fautes de flexion* : mauvaise construction d'un pluriel, d'une forme conjuguée, d'un adverbe...

  exemple :

  - *"il génére(nère) [...]"*

- les *fautes d'accord* : mauvais emploi des marqueurs linguistiques traduisant les liens entre les différents constituants de la phrase.

  exemple :

  ♠ *"[...] dans une() bac [...]"*

Les erreurs dont la détection nécessite de comprendre l'énoncé sont exclues. Ceci explique que nous ne nous sommes pas attardés sur la résolution des anaphores et que nous nous sommes limités aux fautes pouvant être détectées au sein d'une même proposition.

exemple :

♠ *"[...] les coups de l'adversaire qui amène(ent) effectivement au mat."*

Comment savoir sans une certaine compréhension de l'énoncé si le pronom relatif "qui" *se rapporte au nom* "adversaire" *ou au nom* "coups" *et détecter alors une faute d'accord en nombre avec le verbe* "amène".

2.1.2 Règles de bon usage et processus de détection

Le problème posé s'apparente aux problèmes de gestion de processus techniques tels que le diagnostic de pannes ou l'aide à la conduite [Davis 84][Emond 85] : on dispose d'un ensemble de lois qui régissent le bon fonctionnement d'une machine ou d'un système. Compte tenu de ces lois, il existe un ensemble d'anomalies possibles. Dans une application de diagnostic technique, l'objectif est de trouver la *cause* de ces pannes, tandis qu'un système d'aide à la conduite de processus est orienté vers la *prévision* des irrégularités [Sislenchi 85].

Un certain nombre de systèmes de diagnostic technique procèdent en répertoriant les défaillances possibles du système concerné, puis en constituant, à l'aide d'un expert, un ensemble de règles liant défaillances et symptômes. Les symptômes sont les comportements de certains constituants, *observables* et *non attendus* dans un fonctionnement normal. Le raisonnement consiste à rechercher la(les) règle(s) décrivant l'ensemble de symptômes le plus proche de la réalité observée.

Nous aurions pu, de façon similaire, choisir de dresser la liste de toutes les erreurs possibles dans un phrase écrite. Nous avons préféré suivre la démarche de [VandeVelde 85] qui considère le diagnostic comme un mécanisme indépendant et qui opère sur les règles de fonctionnement normal du système.

Nous avons donc conçu notre système en effectuant une distinction nette entre :

- la description de la langue,
- et les mécanismes qui exploitent ces informations afin de détecter les erreurs.

Cette façon de procéder permet d'éviter l'importante étape d'acquisition de l'expertise qui intervenait dans la technique précédente. Elle facilite l'expression des connaissances, qui deviennent celles du bon fonctionnement et qui sont plus facilement accessibles. Elle diminue aussi le risque de redondances et permet une mise au point et des modifications plus commodes. Ce choix apporte une richesse supplémentaire lors de l'explication de l'erreur à l'utilisateur puisqu'on dispose des règles correctes mal utilisées.

Un des objectifs de notre projet étant de gérer les différents mécanismes à l'aide de stratégies à profondeur variable, cette idée est explicitée dans la section suivante.

2.2 Raisonner à Profondeur Variable

Les raisonnements habituellement utilisés en Intelligence Artificielle opèrent la plupart du temps à *profondeur fixe*. En effet, les objets manipulés sont décrits par un ensemble d'informations qui est constant quelle que soit la tâche visée.

En ce qui concerne le traitement des langues naturelles par exemple, ces modélisations tendent à prévoir des représentations canoniques du sens. Une représentation n'est canonique que par rapport à un contexte défini, et dans beaucoup de situations elle devient, soit inexacte, soit inintéressante : si la représentation inclut peu de détails pour un mot, beaucoup de nuances importantes ne sont pas perçues, si elle est trop détaillée, elle est difficile à exploiter. Une différence essentielle entre un raisonnement artificiel et un raisonnement naturel consiste en la faculté qu'ont les humains d'ajuster la profondeur d'analyse à la situation : choix des connaissances à intégrer, finesse des processus qui les exploitent.

La notion de *profondeur variable* est apparue dans les travaux de Bobrow et Winograd [Bobrow 77][Lenhert 79] et de Schank [Schank 79] afin d'exprimer qu'il n'y a pas unicité d'interprétation d'un texte et qu'il est préférable d'adapter la représentation du sens en fonction des objectifs. Schank introduit notamment la notion d'*intérêt* d'une représentation par rapport à une certaine *normalité*. D. Coulon et D. Kayser [Coulon 80]

[Coulon 82] ont formalisé cette démarche et lui ont donné une première réalité informatique. [Kayser 88] et [Castaing 88] ont établi une liaison entre de tels systèmes et les raisonnements non monotones.

Un système qui opère à profondeur variable admet une ou plusieurs des caractéristiques suivantes [Coulon 80]:

- une description progressive des connaissances,
- la pluralité des stratégies,
- une évaluation des approximations possibles.

Les paragraphes suivants présentent l'intérêt de chacun de ces 3 aspects.

2.2.1 La description progressive des connaissances

La difficulté de représentation des connaissances dans un système informatique réside dans la multiplicité des fragments qui la constituent, leur souplesse, leur degré de précision ... Pour que les performances d'un système d'intelligence artificielle restent *raisonnables*, il ne faut pas à une étape donnée noyer le raisonneur avec la totalité de la connaissance disponible concernant chaque objet. Pour éviter ce phénomène, l'utilisateur peut prévoir plusieurs descriptions de ses objets. Il s'agit ensuite de pouvoir sélectionner celle qui est adaptée à la situation courante. Ces descriptions peuvent varier simplement par le niveau de détail qu'elles offrent. On peut ainsi prévoir une *représentation progressive* des objets qui permette l'insertion, de façon graduelle, de connaissances de plus en plus fines.

Pour reprendre l'analogie faite dans [Coulon 82], nous pouvons comparer cette façon de procéder avec la technique utilisée dans les dictionnaires encyclopédiques pour donner l'interprétation d'un mot. L'usage le plus général de ce mot est tout d'abord fourni, ses significations plus particulières sont données ensuite. Ces dernières sont plus détaillées parce qu'elles sont associées à des contextes particuliers et donc moins facilement détectables. La recherche du sens d'un mot s'effectue par touches successives, faisant intervenir progressivement différents niveaux de précision dans l'approche de son environnement.

De façon similaire, une description progressive des objets manipulés dans un système informatique consiste à associer à un concept des caractéristiques, selon un ordre permettant un affinement de sa description de plus en plus poussé. Cela doit permettre de pouvoir raisonner sur les structures les plus simples qui suffisent et sur un nombre minimal

d'entités nécessaires à l'obtention d'un résultat. La description progressive des connaissances concerne d'ailleurs tout aussi bien la description des faits que l'on manipule que celle des inférences que l'on doit conduire.

2.2.2 La pluralité des stratégies

Sans considérer les différents niveaux de détail qu'ils apportent, les points de vue pour appréhender un objet peuvent être multiples. Afin d'illustrer ce propos, prenons l'exemple suivant : un cuisinier perçoit peut-être un plat cuisiné d'abord comme un ensemble d'ingrédients qu'il faut combiner selon un certain minutage. Le gastronome, lui, le considère tout de suite d'un point de vue gourmand : son aspect, son arôme, son goût...

Il ne faut pas disposer d'une stratégie uniforme pour exploiter les propriétés d'un objet. Les caractéristiques du problème traité, le contexte de travail doivent apporter autant d'informations pour permettre de définir des stratégies à suivre adaptées à une étape donnée : choix des facettes intéressantes d'un objet, degré de finesse à exiger ...

Le niveau des connaissances à intégrer peut également être adapté selon les circonstances en faisant intervenir des raisonnements moins rigoureux.

La logique classique ne permet pas de porter atteinte à la validité des raisonnements à cause de la rigueur nécessaire à la déduction logique. On peut penser cependant que, sans nuire outre mesure à la fiabilité du raisonnement effectué, il est permis de prendre parfois des décisions sur des informations approximatives. Pourquoi par exemple, dans certaines situations typiques, ne pas autoriser un moteur d'inférences à appliquer des règles dont on aurait occulté une partie des conditions d'application ? Conditions qu'on aurait choisies, bien sûr, parmi les plus anodines.

Les résultats approchés qui sont fournis peuvent orienter la recherche d'une solution précise. [Karoubi 86] et [David 85] ont suivi cette optique respectivement avec le raisonnement caricatural et le raisonnement par analogie.

De plus, il existe souvent plusieurs façons de résoudre un problème donné [Fox 81]. On peut par exemple déterminer la source d'une panne soit en simulant le fonctionnement détaillé du matériel, soit à l'aide du technicien-expert qui en 20 ans s'est trouvé dans presque toutes les situations de pannes, soit en remplaçant une par une les pièces qui composent la machine. Ces solutions ne sont d'ailleurs pas forcément indépendantes les unes des autres. Le plus intéressant est de pouvoir adapter la stratégie à suivre en fonction des objectifs, des ressources, des informations déjà obtenues ... [Fink 85].

L'idée conductrice du raisonnement à "profondeur variable" est d'adapter le comportement du système aux situations rencontrées. Dans cet objectif, il est souhaitable de ne pas suivre une stratégie figée mais de prévoir une multiplicité de stratégies.

2.2.3 Evaluation des approximations

La mise en œuvre de stratégies à profondeur variable amène à considérer des connaissances qui ne sont pas assurément vraies ou assurément fausses, ou à utiliser des raisonnements qui ne sont pas rigoureux à proprement parler. Les informations manipulées seront alors incertaines, imprécises ou vagues, incomplètes ou évolutives [Chouraqui 85].

Si des systèmes experts connus tels que MYCIN [Farreny 85] ou PROSPECTOR [Lauriere 82] utilisent des modèles très empiriques, d'autres approches offrent des formalismes permettant le traitement de l'imprécision et de l'incertitude s'appuyant sur des modèles mathématiques tels que la théorie des croyances de Shafer, la théorie des possibilités de Zadeh, la théorie des ensembles flous ou la théorie des probabilités. On trouvera dans [Prade 83] un présentation synthétique de ces techniques et les références bibliographiques associées.

Lorsqu'on utilise une multiplicité de stratégies, il faut pouvoir évaluer le risque pris dans l'utilisation d'un raisonnement plus ou moins approximatif, par rapport à un raisonnement plus sûr.

Dans [Sedogbo 83] des résultats de tests sur 10 analyseurs syntaxiques du groupe nominal sont exposés. La description progressive se traduit par l'augmentation successive des contraintes dans les divers analyseurs. La qualité d'un analyseur est apprécié en mesurant son taux de confiance (tc) :

$$tc = echo/(echo + silence + bruit)$$

L'écho, le silence et le bruit correspondent respectivement au nombre de chaines correctement analysées, oubliées ou mal interprétrées, eu égard au modèle de référence.

La complexité (cp) est plus sommairement estimée en fonction de l'encombrement mémoire de l'analyseur, la taille de la matrice de précédences catégorielles utilisée et enfin le nombre d'instructions de comparaison exécutées au cours de l'appel d'un sous-programme de vérification.

Les rapports *qualité-complexité* de chacun d'entre eux sont reportés sur une courbe. Parler de profondeur variable revient à établir un ordre sur cette courbe qui permet d'ajuster la profondeur choisie en fonction de la qualité attendue et des ressources al-

louées. Cette expérience montre que la profondeur peut être estimée au moyen de critères objectifs.

* * *

Suite à ces réflexions, on ressent de façon intuitive l'intérêt général de raisonnements à profondeur variable. Il s'agit de s'interroger maintenant sur l'opportunité d'intégrer cette notion dans un système de détection de fautes d'orthographe. C'est ce que nous nous proposons de faire dans le paragraphe suivant.

2.3 Raisonner à profondeur variable en détection de fautes d'orthographe

Pour détecter les fautes dans un énoncé, il faut schématiquement : vérifier que chaque mot est bien construit, reconnaître la structure syntaxique de cet énoncé pour contrôler les règles de construction qu'elle fait intervenir.

Nous décrivons ci-dessous quelques situations caractéristiques qui peuvent apparaître au cours de la réalisation de l'une ou l'autre de ces tâches. Est mise en relief la variété des traitements envisageables et des niveaux de précision qu'ils offrent. Quelques exemples montrent l'utilité de la diversité et de la progressivité des connaissances pour guider la stratégie d'analyse.

2.3.1 Multiplicité des traitements et des solutions

exemple :

Plusieurs erreurs ont été volontairement insérées dans la phrase suivante.

♠ *"L'un des bateaus(x) était tiré par un chalutier de la marine royale."*

Cet exemple va permettre d'illustrer la variété des traitements possibles pour résoudre un même problème. La nature des résultats obtenus, leur précision devraient permettre de déterminer celui qui convient à l'objectif recherché. Considérons le mot erroné *"bateaus"*. La faute peut être détectée par une recherche dans un lexique contenant tous les mots français ainsi que leurs formes fléchies. Cette forme y est absente et donc a priori mal écrite. Mais on ne dispose pas toujours d'une connaissance aussi exhaustive. Dans l'exemple, une simple étude des désinences autorisées des mots français suffit à dénoncer la faute : *"-eaus"*

n'est pas une terminaison autorisée. Ces deux solutions offrent simplement une vérification des chaînes de caractères, aucune n'apporte d'informations sur une éventuelle correction.

Pour effectuer des hypothèses sur l'origine de la faute il est intéressant d'observer le mot dans son contexte syntaxique. Sa place dans la phrase, l'étude de ses voisins grammaticaux incite à prédire un nom. Dans ce cas, le respect de l'accord en nombre avec l'article contracté *"des"* qui le précède peut compléter l'hypothèse en proposant de chercher un nom pluriel. En consultant les lois de formation du pluriel des noms, on remarque que *"bateaus"* est une forme dérivée de *"bateau"*. La règle appliquée serait la règle la plus générale qui construit le pluriel d'un nom en lui ajoutant un *"-s"*. On complète le diagnostic en constatant que la règle exception sur le pluriel des noms en *"-eau"* n'a pas été prise en compte. On peut même proposer une correction de *"bateaus"* en *"bateaux"*.

Cette redondance des traitements incite donc à réfléchir au préalable sur celui qui convient le mieux en fonction des objectifs recherchés, des informations et des ressources disponibles. Ce choix n'étant pas unique, il convient de doter le système de la possibilité de *réfléchir* sur l'opportunité d'une stratégie. L'étude seule de la terminaison permet seulement d'émettre un doute sur la validité du mot *"bateaus"*. La mise à jour de la règle sur le pluriel des noms en *"-eau"* certifie l'erreur. Elle permettra en plus de proposer une correction et d'aider à sa compréhension.

De façon générale, selon l'objectif fixé, on pourra se satisfaire d'une lecture rapide pour détecter les fautes les plus grossières, ou bien d'une étude un peu plus approfondie permettant de suggérer des corrections grâce à l'extraction de la règle mal utilisée.

2.3.2 Connaissances incomplètes et recoupement

L'ensemble des constructions possibles de phrases en langage naturel est immense. Un système de détection de fautes devrait être apte à reconnaître la structure du plus grand nombre possible de phrases. Or dans tout système, une connaissance exhaustive devient difficile à gérer : recherche des informations pertinentes, maintien de la cohérence ... Plutôt que concentrer une connaissance trop riche d'un côté, il est possible de procéder dans certains cas par recoupement d'informations de sources diversifiées.

Dans l'exemple précédent, *"L'un"* est une tournure particulière qui se justifie par un effet de prononciation. A une connaissance complète de la syntaxe qui intégrerait cette structure, il est préférable de juxtaposer une connaissance partielle de la syntaxe et une information sur des notions phonétiques.

L'arbre syntaxique de la figure 2.1 page 37 décrit une interprétation syntaxique d'une partie de la phrase. La règle suivante semble compléter cet arbre et permettre la reconnaissance de la phrase :

∇ *Le pronom* "un" *construit avec un complément peut être précédé de l'article élidé* "l'" *[Grevisse 75].*

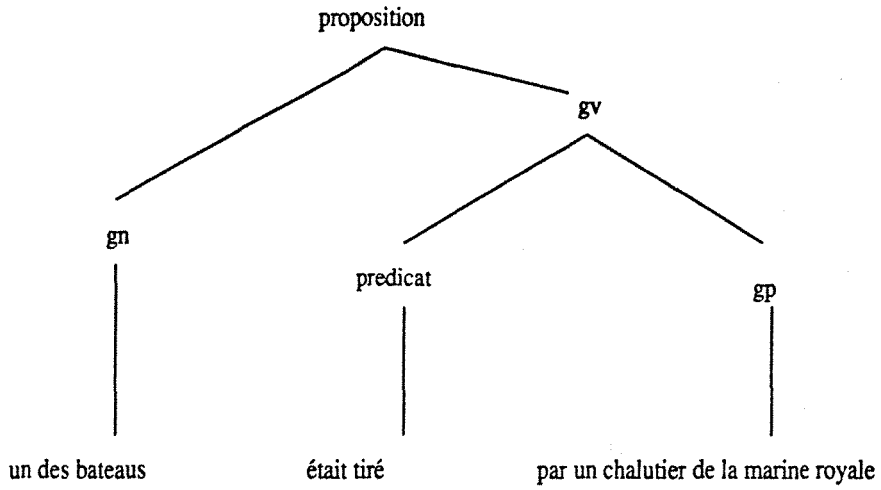


Figure 2.1.

2.3.3 Raisonnement approximatif mais pertinent

Pour détecter le plus grand nombre de fautes possibles ou pour obtenir des informations supplémentaires afin de mieux cerner l'erreur, il est intéressant de ne pas arrêter l'analyse d'une phrase dès l'apparition d'une anomalie. Une solution consiste à corriger de façon automatique toute faute rencontrée. Une autre réside en une analyse tolérante aux fautes.

Une analyse tolérante revient à accepter des hypothèses obtenues par des raisonnements moins rigoureux. La vérification de certaines conditions est repoussée à une étape ultérieure dans laquelle les résultats seront approfondis. Par exemple, la forme fléchie *"bateaus"* n'existe pas en français. Mais la désinence *"-s"* de ce mot peut désigner une marque du pluriel. En effet la règle générale sur le pluriel des noms est :

∇ *Les noms prennent un "-s" au pluriel*

Un premier raisonnement superficiel sans consultation des règles exceptions fournit l'hypothèse (*"bateaus" nom pluriel*). Cette hypothèse est confirmée par le succès de l'analyse de la phrase. Les premières conclusions sont approfondies en tenant compte des règles plus précises décrivant les exceptions. La faute de construction est ainsi détectée.

Schématiquement, cette démarche revient :

- à prévoir la faute au niveau local,

- à effectuer une correction a priori de cette faute, ou à inhiber les contraintes non vérifiées,

- à poursuivre l'analyse après cette correction,

- puis à vérifier la cohérence et l'exactitude des hypothèses effectuées.

Autoriser de telles approximations évite le rejet total de la forme erronée. Cela évite dans d'autres situations de s'engager dans des vérifications inutiles comme retrouver que *"bateau"* au pluriel prend un *"-x"* dans une phrase où il ne doit pas prendre de marque du pluriel, par exemple :

♠ *Le bateau<u>s</u>() était tiré par une() chalutier*

2.3.4 Sélection des résultats

Nous nous sommes tous déjà trouvés en situation de *veille* : conduite d'une voiture, réalisation d'un plat cuisiné... Débutants, tous les stimuli nous font réagir. Avec l'expérience, ils sont maîtrisés, seuls les plus pertinents et les plus inhabituels éveillent en nous une attention particulière.

Il en est de même dans un système de détection d'anomalies. Il ne s'agit pas d'alarmer l'utilisateur intempestivement, mais de sélectionner les signaux à lui transmettre. L'ambiguïté de la langue naturelle, surtout si nous faisons intervenir peu de sémantique, accentue cette difficulté, comme l'illustrent les exemples qui suivent.

Le groupe de mots :

"un chalutier de la marine royale"

n'admet a priori qu'une interprétation syntaxique (figure 2.2) dans laquelle l'adjectif *"royale"* se rapporte au nom *"marine"*.

Une autre structure syntaxiquement reconnue modulo la faute d'accord en genre entre l'adjectif *"royale"* et le nom *"chalutier"* est décrite en figure 2.3. Si aucun traitement complémentaire ne permet de rejeter cette solution, pour des raisons sémantiques par exemple,

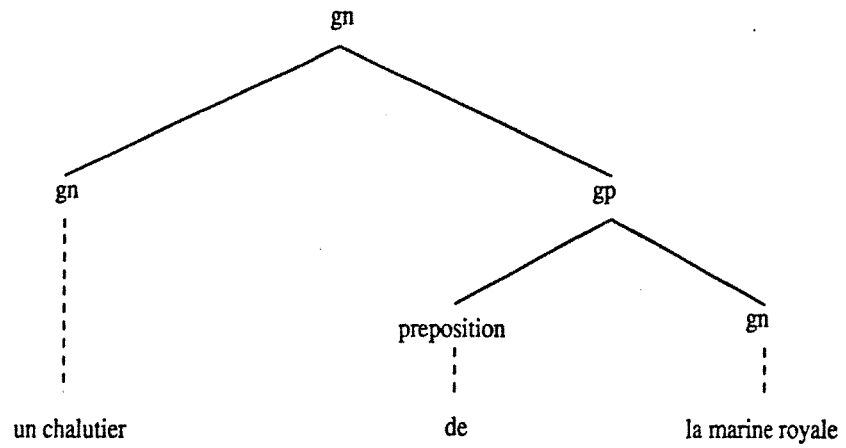


Figure 2.2.

un système de détection de fautes d'orthographe devrait a priori envisager également cette possibilité. il serait nécessaire alors de signaler la faute d'accord dans le groupe de mots *"chalutier de la marine royale()"*.

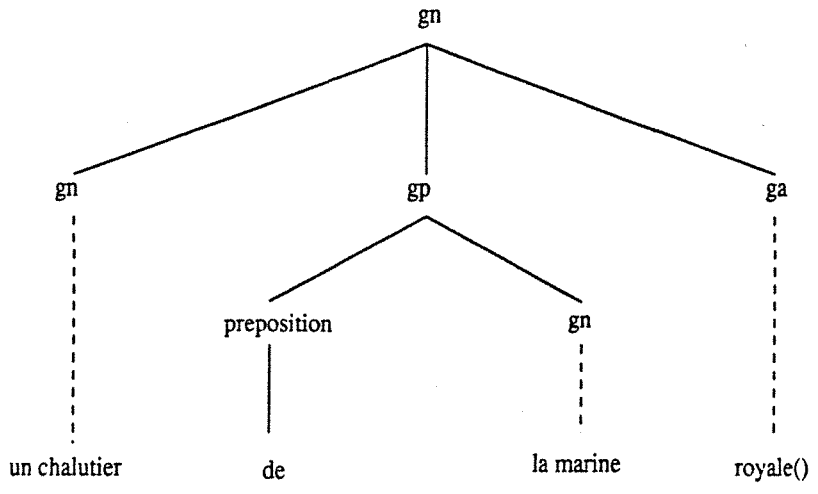


Figure 2.3.

Plusieurs attitudes sont possibles pour un système de détection de fautes :

- accepter toutes les interprétations et signaler à l'utilisateur celles qui contiennent des anomalies,

- favoriser l'une d'entre elles, et transmettre le message adapté.

Il ne semble pas raisonnable et peu ergonomique de signaler systématiquement à l'utilisateur toutes les solutions reconnues avec ou sans erreur. Dans le corpus que nous avons constitué (cf annexe), rassemblant des articles de journaux, des articles scientifiques et des dictées d'enfants, nous avons recensé un nombre négligeable (moins de 1%) de groupes syntaxiques simples dans lesquels se trouvent ambiguïté d'accord et qui comportent une erreur. Il semble plus judicieux de favoriser la solution qui ne contient pas de fautes et de passer sous silence les autres interprétations. Le risque pris est jugé faible.

Dans d'autres situations, il est trop permissif d'accepter la solution correcte au bénéfice du doute. C'est le cas par exemple quand cette solution fait référence à des règles d'accord d'usage rare.

exemple : Accord de l'adjectif avec deux ou plusieurs noms coordonnés.

Il est raisonnable de se demander s'il faut écrire :

forme 1 : " l'organisation et le rôle clandestin des moudjahiddin"

ou bien

forme 2 : " l'organisation et le rôle clandestins() des moudjahiddin"

Il ne faudrait pas systématiquement le faire pour des situations similaires :

" ... négocier les salaires et les modifications mineures."

Il faut trouver un compromis à ces deux solutions. Il est certainement plus sage de forcer l'application de la règle la plus courante qui rapporte l'adjectif au nom le plus proche, quitte à signaler des erreurs non justifiées dans un faible nombre de cas, comme la marque *"s"* du pluriel sur *"moudjahiddin"* dans la seconde forme possible. Doser le mieux possible le *bruit*, surplus de fautes détectées, et le *silence*, omission de fautes et concevoir les stratégies en conséquence revient à définir la précision des résultats à signaler à l'utilisateur.

2.4 Une analyse progressive

Nous venons d'illustrer avec quelques exemples des situations pour lesquelles il nous semble enrichissant de moduler la profondeur des traitements appliqués et d'étudier les

effets de raisonnements approchés : utilisation de mécanismes plus ou moins sophistiqués, exploitation de connaissances de façon approximative...

Dans les chapitres suivants nous montrons comment nous avons implanté cette approximation. Nous avons en premier lieu prévu une description progressive des connaissances de tout niveau. Les informations sont organisées de manière à connaître le plus général avant le particulier. Nous avons ensuite permis l'exploitation de ces connaissances à des niveaux de précision et de rigueur différents. Nous avons notamment fourni le moyen d'implanter un processus général d'*analyse progressive*. Ce mécanisme permet de cerner dans un premier temps la solution par des raisonnements sur des informations générales et grossières. Les résultats sont affinés progressivement en faisant intervenir des connaissances de plus en plus précises, complètes ou adaptées au contexte. On continue jusqu'à atteindre un niveau de détail considéré comme suffisant pour valider les résultats obtenus.

Pour décrire de façon simplifiée la coopération entre les différentes sources d'informations, les degrés de finesse et de complexité des traitements déclenchés par les stratégies, nous avons défini pour ces dernières une écriture déclarative. Des stratégies ont été développées pour la détection des fautes d'orthographe en utilisant ce formalisme. On a pu ainsi tester son applicabilité.

Après avoir décrit les formalismes choisis pour exprimer connaissances de la langue et stratégies, nous présenterons notre expérimentation et ses résultats.

3
Représentation des règles du français

Le sujet de la représentation des connaissances recouvre plusieurs problèmes. On peut citer les questions soulevées par le formalisme de la représentation, par les méthodes et les moyens d'accès aux connaissances, par leur modification et la production de nouvelles connaissances [Farreny 87], il faut aussi se demander que représenter, quelle richesse introduire dans cette représentation ? Dans notre système, la distinction est faite entre les règles de bon usage et les mécanismes de détection de fautes et ce chapitre est consacré aux formalismes utilisés pour décrire les connaissances sur la langue.

Les exemples développés dans le chapitre précédent ont pu donner une idée de la démarche générale que nous souhaitons suivre dans l'analyse d'une phrase. La représentation des connaissances a été choisie pour cadrer au mieux avec le principe d'analyse à profondeur variable ; un formalisme homogène (règles) a été prévu pour permettre des raisonnements progressifs. Ces règles sont organisées pour aller du plus général au plus particulier, et leurs conditions d'application sont classées selon leur importance.

3.1 Organisation générale des connaissances

La description de la langue donnée dans les livres de grammaire rassemble des *fragments* de connaissances de natures linguistiques différentes : morphologique, syntaxique, sémantique..., mais aussi de niveaux différents : règles générales, listes d'exceptions, de cas particuliers ... Ils correspondent à des unités de savoir-faire qui associent à des classes de situations établies, ou à établir, les conséquences à tirer ou bien les actions à accomplir.

exemples[1] :

∇ *Si je veux construire le pluriel d'un nom,*
alors j'ajoute un "-s" à la fin de ce nom.

∇ *Si il existe un déterminant suivi d'un nom,*
alors ils peuvent former un groupe nominal
et je dois vérifier qu'ils ont même genre et même nombre.

Nous avons choisi le formalisme des règles de production qui nous semble bien adapté pour transcrire ces granules de connaissances et pour mettre en œuvre un raisonnement à *profondeur variable* autorisant des raisonnements aproximatifs. Deux idées directrices ont été retenues pour définir la structure générale des règles. D'une part elle doit, sous certaines conditions, permettre l'application d'une règle dont la validation est entachée d'incertitude. Nous verrons en effet que certaines conditions ne sont pas toujours vérifiées ou vérifiables. Encore faudra-t-il distinguer les cas où l'impasse effectuée n'est pas préjudiciable pour la suite des traitements. D'autre part, lorsqu'une connaissance grossière s'avère insuffisante, cette structure doit faciliter l'accès aux informations plus fines qui décrivent le même phénomène.

En résumé, la structure générale des règles du français doit prévoir une *hiérarchie* des connaissances selon leur degré de finesse. Elle inclut aussi un *partitionnement des conditions* d'application des règles selon leur rôle et leur importance (figure 3.1).

Règle du français	
Ensemble	*des conditions d'activation de la règle décrivant son contexte d'application*
Ensemble	*des conditions dont la vérification peut être retardée ou occultée à certaines étapes*
Ensemble	*des conclusions à tirer*
Liens	*vers des connaissances plus fines*

Figure 3.1. Structure générale des règles sur le fonctionnement de la langue

1. Les règles du français données en exemple sont inspirées de manuels dont les références sont notées dans notre texte en italiques (*[reference]*).

Nous allons étudier comment se traduisent ces hypothèses de travail sur les différentes classes de connaissances : morphologie, syntaxe ...? Peut-on les suivre toutes ? Comment vont-elles s'exprimer ?

Il n'est pas question dans ce chapitre de décrire toutes les classes de connaissances que les stratégies peuvent faire intervenir dans l'analyse d'un énoncé : phonétique, statistique, etc... Seul un noyau minimum d'informations indispensables est étudié ici pour l'objectif fixé : détecter les mots inconnus de la langue et déceler en particulier les fautes de flexion et certaines fautes d'accord entre les constituants de l'énoncé.

Ce noyau comprend :

- un lexique à partir duquel retrouver les mots autorisés en français ;
- les règles de fléchissement morphologique des mots (formation du pluriel, du féminin, des conjugaisons ...) appelées *règles morphologiques* ;
- des informations ou *connaissances syntaxiques* pour repérer les groupes de mots (syntagmes) dans lesquels il y a des accords à respecter.

Les représentations définies pour exprimer ces connaissances sont décrites dans les sections 3.2, 3.3 et 3.4. Les formalismes choisis sont homogènes. Il sera facile de les utiliser pour décrire des informations complémentaires dans d'autres domaines, éventuellement introduits par les stratégies.

3.2 Le lexique

Structure générale et informations stockées

Une façon de savoir si un mot appartient à la langue considérée, est de le rechercher dans un lexique des mots autorisés. S'il ne s'y trouve pas tel quel, il est possible de le retrouver à une variante morphologique près. Le lexique contient alors les mots grammaticaux (déterminant, adverbe ...) et les formes irrégulières dont on connaît la liste, les mots invariables et les radicaux (éléments irréductibles : adjectif masculin singulier, verbe infinitif...) des formes variables régies par les règles de flexion. Cette solution que nous avons choisie a l'avantage de diminuer la taille du lexique et de fournir l'ensemble des lois à respecter, il augmente cependant les traitements à effectuer pour retrouver une forme.

Le lexique fournit pour chaque entrée[2] des renseignements de type grammatical. Ils sont nécessaires pour retrouver les lois morphologiques qui régissent le fléchissement d'une racine et vérifier les contraintes d'utilisation d'un mot dans une phrase : règles de combinaison syntaxique, contraintes d'accord.

exemples :

"doux"	*adjectif masculin singulier*
"lancer"	*verbe infinitif 1er groupe*
	nom masculin singulier
"munition"	*nom féminin singulier*
"nouveau"	*adjectif masculin singulier*
	adverbe
	nom masculin singulier
"par"	*préposition*

On appelle *hypothèse grammaticale* une liste des couples *(attribut valeur)* qui caractérisent un mot utilisé dans une phrase.

exemple :

((nature adjectif)(genre masculin)(nombre singulier))

Selon le contexte syntaxique dans lequel il est employé, un mot peut parfois être caractérisé par des hypothèses grammaticales différentes repérées par un numéro[3]. Chacune de ces possiblités apparaît dans le lexique car le mot y est considéré hors de tout contexte syntaxique. A charge des analyses de déterminer l'hypothèse correspondant à la situation.

exemple :

("lancer" (1 (nature nom)(genre masc)(nombre sing))
(2 (nature infinitif)(groupe 1)))

Voici la grammaire du lexique utilisé dans notre réalisation. Elle est donnée sous une forme de Backus-Naur dans laquelle les non-terminaux sont notés en majuscule.

2. Une entrée correspond pour le moment à un mot. Même s'il est évident que la prise en compte des locutions, des expressions figées etc... est indispensable, nous ne l'avons actuellement pas intégrée.

3. Dans la phase de développement du projet, nous n'avons pas étudié de façon approfondie la structure du lexique. Nous remettons à la phase d'intégration d'un module de recherche de voisins des mots mal orthographiés, le choix d'un dictionnaire plus performant.

LEXIQUE	:=	*ENTREE-LEXICALE **
ENTREE-LEXICALE	:=	*"(" ENTREE LISTE-HYPOTHESES ")"*
ENTREE	:=	*CHAINE*
LISTE-HYPOTHESES	:=	*HYPOTHESE **
HYPOTHESE	:=	*"(" NUMERO LISTE-PROPRIETES ")"*
LISTE-PROPRIETES	:=	*PROPRIETE **
PROPRIETE	:=	*"(" PREDICAT VALEUR ")"*
PREDICAT	:=	*nature / genre / nombre / groupe / personne / temps*
VALEUR	:=	*nom / verbe / infinitif / det / propers / adj / ...*
CHAINE	:=	*LETTRE **
LETTRE	:=	*a/b/c/...y/z*
NUMERO	:=	*1/2/3...*

exemple :

Lexique = (... *(* "jouer" *(1 (nature infinitif)(groupe 1)))*

...

("nouveau" *(1 (nature adj)(genre masc)(nombre sing))*
(2 (nature nom)(genre masc)(nombre sing))
(3 (nature adverbe)))

...

("par" *(1 (nature prep))) ...)*

Un lexique à deux ou trois niveaux

Dans [Catach 84b], Nina Catach liste 158 mots et formes fléchies de la langue française qui couvrent environ 60% des occurrences dans tous les textes courants, et 4000 formes qui couvrent en moyenne 90% des occurrences d'un texte. Ces observations ont inspiré des lexiques à plusieurs niveaux [Peterson 80] : un petit lexique des mots les plus fréquents de la langue, et un lexique plus grand contenant les autres mots de la langue. On peut également prévoir un lexique personnalisé contenant par exemple des noms propres ou des termes techniques, ou bien des mots dont l'orthographe fait souvent buter l'utilisateur.

3.3 Les connaissances morphologiques

La morphologie est l'étude de la formation des mots et des variations de forme qu'ils peuvent subir : singulier et pluriel, masculin et féminin, dérivés, composés, etc ...

exemples :

∇ *Le pluriel de* "carnaval" *est* "carnavals"

∇ La table de conjugaison au présent de l'indicatif du verbe aimer est :

| j'aime
| tu aimes
| il aime
| nous aimons
| vous aimez
| ils aiment

∇ *L'adverbe se forme en ajoutant* "-ment" *au féminin de l'adjectif*

[Gruaz 86] propose de réduire encore le nombre des formes de base en prenant comme unité non pas le mot (défini comme la forme linguistique minimale douée de sens), mais le morphème qui est le composant minimal porteur de sens interne du mot (*exemple :* terr(e) dans terre, terrain, terrien, terrier). Cette décomposition qui nécessite la prise en compte de phénomènes assez complexes n'a pas été intégrée. Seules les règles qui décrivent les variations des mots relatives aux différents changements des attributs grammaticaux : genre, nombre, conjugaison... ont été considérées.

3.3.1 Structure des règles

Généralités

Les descriptions des variations morphologiques des mots sont relativement homogènes dans tous les ouvrages de grammaire. La règle complète sur la formation du pluriel des noms s'exprime par exemple sous des formes analogues à celle donnée dans la figure 3.2.

De façon générale, les règles sur la morphologie sont conçues dans les manuels pour régler le problème de la construction de formes fléchies. Dans notre système, pour respecter la démarche naturelle de l'expert, et même si elles doivent servir à d'autres buts, elles ont été définies pour ce type de problèmes. Une utilisation et des stratégies adaptées (cf chapitre 4) permettront facilement de les exploiter pour retrouver si une forme existe dans la langue et déceler les irrégularités qui apparaissent dans leur utilisation.

Le pluriel des noms

∇ **R1** *On forme le pluriel des noms en ajoutant un "-s" au singulier : un homme, des hommes.*
exceptions *:*
∇ **R1.1** *Les noms terminés par "-s", "-x", "-z", ne changent pas au pluriel : une croix, des croix.*
∇ **R1.2** *Les noms en "-al" forment leur pluriel en "-aux" : un journal, des journaux.*
exceptions *:*
∇ **R1.2.1** *Cependant les noms suivants ont leur pluriel en "-s" : bal, cal, carnaval,chacal, festival, regal.*
∇ **R1.2.2** *Il faut ajouter les noms moins usités : cérémonial, santal ...*
exceptions *[Grevisse 75] :*
∇ **R1.2.2.1** *Les pharmaciens disent : "poudre des trois santaux".*
∇ **R1.2.3** *Les noms suivants ont deux formes possibles au pluriel, en "-s" ou en "-aux" : "étal", "final" ...*
∇ ...
∇ ...

Figure 3.2.

Elles présentent toutes un ensemble de propriétés communes :

- Elles sont données par unités de savoir-faire et elles traduisent en général des implications.
- Ces unités sont données de façon graduelle. La règle qui décrit le comportement à suivre dans la plupart des situations est donnée tout d'abord (R1.2), les règles qui décrivent le comportement à adopter dans des situations plus particulières sont décrites ensuite (R1.2.1 R1.2.2 R1.2.3...). Dans de telles situations, la règle plus générale ne doit pas être appliquée, on peut donc parler d'une hiérarchie d'**exceptions**.
- Une règle peut exprimer également un choix entre deux comportements possibles (R1.2 ou R1.2.3).

- Elles sont données par ordre d'**importance**. On en rencontre dont la rareté d'usage ou la spécificité du domaine d'application offre un intérêt limité (R1.2.2.1).
- Ces lois sont **incomplètes**. Elles ne précisent pas comment sont propagés les attributs syntaxiques d'une forme vers une autre. Par exemple, il est sous-entendu qu'un nom pluriel garde le même genre que le nom singulier dont il est dérivé.

La forme progressive qu'on souhaite intégrer dans la représentation des connaissances apparaît naturellement dans les lois énoncées par les experts. La structure choisie pour les règles morphologiques du système reproduit cette progressivité, et ces dernières restent très proches de celles des manuels de grammaire.

Structure

Une *règle morphologique* décrit une unité de savoir-faire (R1, R1.2, R1.2.1, ...) concernant le fléchissement des mots indépendamment de toute contrainte syntaxique. Elle est composée d'un certain nombre de conditions à remplir pour pouvoir être appliquée et de conclusions que l'on peut déduire si ces conditions sont vérifiées. Ces conclusions ne se limitent pas à la description d'une nouvelle forme fléchie, en effet, certaines informations apparaissant implicitement dans les règles des manuels doivent être explicites dans les règles morphologiques pour pouvoir être exploitées par le système. Elles portent entre autres sur les attributs grammaticaux qui caractérisent la forme construite et le radical dont elle est dérivée. On obtient ainsi pour la règle R1.2 la forme suivante :

exemple :

∇ *Si un mot s'écrit x-*"al"
et si c'est un nom
Alors on peut déduire a priori que le mot écrit x-"aux"
peut être un nom pluriel de même genre que x-"al"
(sous réserve des exceptions).

Le niveau de généralité d'une règle lui confère un certain degré de finesse. Une règle d'exception opère en effet sur des informations plus précises que la règle qu'elle complète et on peut dire qu'elle constitue de ce fait une connaissance plus fine. Pour exprimer cet affinement progressif des informations, chaque règle est pourvue de la liste de ses règles d'exceptions. On fournit ainsi la possibilité de faire varier le niveau de détail des règles

consultées. On perd en contrepartie l'avantage d'avoir des règles indépendantes les unes des autres.

Les conditions d'application d'une règle ont des rôles variés. On distingue celles qui portent sur la forme des mots : désinence, liste de mots ... (*exemple* : si un mot se termine par *"-al"*) qui permettent une sélection rapide de la règle. Elles traduisent en plus la discrimination entre une règle et ses règles exceptions. Les autres conditions précisent le contexte de validité de la règle, par exemple les attributs grammaticaux des mots qu'elle fait varier.

La conclusion énoncée par la règle concerne le mot en tant qu'élément de la langue (par opposition au *mot de la phrase*) et a valeur d'hypothèse puisque plusieurs conclusions contradictoires peuvent être apportées par des règles différentes (par exemple *"guides"* est reconnue comme une forme conjuguée de *"guider"* ou comme la forme pluriel du nom *"guide"*). Le passage des informations du *mot de la langue* au *mot de la phrase* est du ressort des stratégies d'analyse mises en œuvre (cf chapitre 5).

La structure générale des règles morphologiques est donnée figure 3.3. Chaque ensemble d'informations est introduit par un mot-clé déterminé.

SI	*"conditions d'activation rapide de la règle"*
ET_SI	*"conditions précisant le contexte d'application"*
ALORS	*"conclusions admises sous réserve qu'aucune règle exception ne puisse s'appliquer."*
SAUF	*"liste des règles contredisant les conclusions pour les différentes exceptions."*
OU	*"liste des règles pouvant être appliquées pour le même phénomène."*

Figure 3.3. *Forme générale des règles de morphologie*

La clause *OU* permet de préciser les situations pour lesquelles il y a plusieurs solutions possibles pour un même contexte, solutions entre lesquelles on ne sait pas a priori choisir.

Lorsqu'une règle R_i renvoie à une règle *Ou* R_j , seul ce lien entre les deux règles existe. Il n'y a pas de lien *Ou* de R_j vers R_i. La règle d'entrée (R_i) est la plus générale des deux règles, par exemple on a: *R1.2* OU *R1.2.3.*

Les formalismes pour exprimer conditions, conclusions et liste de règles sont détaillés dans les paragraphes suivants.

3.3.2 Le formalisme choisi

Une règle morphologique est désignée par le mot-clé *init_morpho*, par un numéro et par ses différentes clauses. Voici la grammaire du langage utilisé pour la décrire.

```
REGLE :=  "("  init_morpho NUMERO
               CLAUSE_SI
               CLAUSE_ET_SI
               CLAUSE_ALORS
               CLAUSE_SOUS_RESERVE
               CLAUSE_SAUF
               CLAUSE_OU ")"
```

a) La base de travail

L'utilisation d'une règle ne peut se décrire que par rapport à un état courant de l'analyse, pour lequel on aura recensé les informations qu'on peut utiliser. A une étape donnée du raisonnement un certain nombre d'hypothèses, parfois concurrentes, ont été inférées sur les constituants de la phrase. L'ensemble de ces hypothèses est contenu dans ce qui est appelé la *base de travail*. C'est dans cette mémoire, figée pendant l'évaluation d'une règle, que sont puisées les informations pour la vérification des conditions, et que sont ajoutées les inférences obtenues après application des règles. Les hypothèses sont stockées dans la base de travail sous le même formalisme que celui utilisé pour ranger les informations du lexique.

exemple :

(...("valses" (1 (nature nom)(genre fem)(nombre plur))
(2 (nature verbe)(personne 2)(nombre sing)(temps present)))...)

b) Les conditions

Expression des conditions

Les conditions sont contenues dans *Clause_Si* et *Clause_Et_Si*. Le langage utilisé pour les traduire s'appuie sur le formalisme des prédicats du premier ordre. Il est décrit ci-

dessous par un vocabulaire, une syntaxe et une sémantique.

Vocabulaire Le vocabulaire comprend :
- des noms de prédicats (*exemples :* nature, groupe ...)
- des noms de procédures externes (*exemples :* unif, appartient ...)
- des noms de variables locales à la règle commençant par un "?" (*exemples :* ?x, ?y ...)
- des noms de variables globales (*exemples :* mot, phrase ...)
- des constantes chaînes de caractères (*exemples :* "aux", "carnaval" ...)
- des constantes littérales (*exemples :* nom, sing ...)
- des constantes entières.

Syntaxe Les termes employés sont :

- des *termes-relations* (R_i a_1 a_2) où les a_j sont des variables ou des constantes, les R_i des noms de prédicats d'arité 2 dont les valeurs sont à retrouver dans les informations de la base de travail.

  exemples :

  (nature nom ?x)

  (groupe 2 mot)

- des *termes-fonctions* (C_i $a_1 ... a_n$) où les a_j sont des variables où des constantes et les C_i des noms de fonctions externes à n arguments.

  exemple :

  (appartient L_PLUR_AL ?x)

  avec L_PLUR_AL = ("bal", "cal", "cantal",...)

Sémantique Chaque terme est évalué en fonction du **contexte de validation** de la règle, construit au fur et à mesure de la vérification de ses conditions et défini par :

- l'*ensemble des instanciations* des variables, déjà effectuées par les vérifications précédentes.

- l'*ensemble des hypothèses* de la base de travail pour lesquelles la règle a été partiellement validée. Une hypothèse est repérée par un couple *(constituant no). constituant* est le constituant de la phrase sur lequel porte l'hypothèse et *no* le numéro de cette hypothèse parmi celles que contient la base de travail sur *constituant.*

exemples :

- *Si la variable* ?x *est déjà instanciée avec la chaîne* "valses" *et la variable* ?g *avec la valeur* fem, *l'*ensemble des instanciations *des variables est :*

((?x "valses")(?g fem))

- *Reprenons les hypothèses sur* "valses" *:*

(..."valses" (1 (nature nom)(genre fem)(nombre plur))
(2 (nature verbe)(personne 2)(nombre sing)(temps present)))...)

Si les premières conditions de la règle ont été validées pour l'hypothèse grammaticale 1 sur le mot "valses", *l'*ensemble des hypothèses *du contexte de validation contient le couple* ("valses" . 1). *Cela revient à travailler avec l'ensemble des informations :*

("valses" (1 (nature nom)(genre fem)(nombre plur)))

- Vérifier un *terme-fonction* consiste à vérifier que le résultat de son exécution rend vrai. Lorsqu'il opère sur des variables non encore instanciées, le déclenchement d'un *terme-fonction* peut compléter le contexte de validation de la règle.

exemple :

La fonction (unif var exp) *teste si la valeur de la variable* var *s'unifie avec l'expression* exp *qui contient des variables et des chaînes de caractères. Si* var *n'a pas de valeur, la fonction* unif *l'instancie avec la valeur de* exp.

Le terme-fonction T = (unif mot (conc ?x "au")) *teste l'unification de* mot *avec une chaîne terminée par* "au" *(la fonction* conc *représente la fonction de concaténation de chaîne).*
Si mot = "chapeau", *l'évaluation de* T *rend vrai et fournit l'instanciation de la variable* ?x *avec la chaîne de caractères* "chape" *: (?x* "chape"*).*

- Un *terme-relation* $T = (R_i\ a_1\ a_2)$ est évalué de la façon suivante :

  - Si le contexte de validation contient déjà une hypothèse sur a_2 on recherche dans la base de travail la valeur V du prédicat R_i pour cette hypothèse.

* Si a_1 est une constante ou une variable instanciée, l'évaluation de T revient à comparer V et a_1.

  exemple :

  Soit dans la base de travail les hypothèses suivantes :

  (...("voyage" (1 (nature nom)
  (genre masc)
  (nombre sing)))
  ("interne" (1 (nature adj)
  (nombre sing))
  (2 (nature nom)
  (nombre sing)))...);

  le terme-relation (genre masc "voyage") *est faux.*

* Si a_1 est une variable non instanciée, l'évaluation de T rend vrai. Le contexte d'évaluation de la règle est complété par l'instanciation de a_1 avec la valeur V.

  exemple :

  Si on prend la même base de travail que ci-dessus et le contexte de validation défini par la liste d'instances
  ((?x "interne")) associée à la liste d'hypothèses de travail (("voyage"
  . 1));
  le terme-relation (nombre ?nb "voyage") *est évalué à vrai.*
  La variable ?nb *est instanciée par* sing *(valeur de l'attribut* nombre *dans l'hypothèse 1 sur* "voyage"*). La liste d'instanciations du contexte de validation devient : ((?x "interne")(?nb sing)) .*

– Si le contexte de validation ne contient aucune hypothèse sur le mot a_2, l'évaluation de T rend autant de contextes de validation que d'hypothèses qui existent sur a_2 dans la base de travail. S'il n'en existe pas, elle rend *"en_attente"* ce qui signifie qu'on ne peut rien dire.

  exemples :

  Toujours avec les mêmes bases de travail et contexte de validation, l'évaluation de (nature ?n ?x) *déclenche la recherche des instanciations possibles pour* ?n . *On trouve* (?n adj) *pour l'hypothèse 1 sur* "interne"

et (?n nom) *pour l'hypothèse 2 sur* "interne". *Deux contextes de validation sont alors rendus :*
(((?x "interne")(?n adj)) (("interne" . 1)))
et (((?x "interne")(?n nom)) (("interne" . 2))).

En résumé, une condition est dite **vérifiée** lorsque :

- exprimée sous forme de relation, elle est unifiable avec une information stockée dans la base de travail ;

- exprimée sous forme de fonction, elles est évaluée vraie.

Les clauses SI et ET_SI

Clause_Si décrit une liste ordonnée de conditions appelées *filtres*. *Clause_Et_Si* complète les conditions de déclenchement de la règle et son environnement d'application.

Voici la grammaire du langage utilisé.

CLAUSE_SI	:=	*"(" "SI" LISTE_CONDITIONS ")"*
LISTE_CONDITIONS	:=	*CONDITION **
CONDITION	:=	*"(" PREDICAT VALEUR VARIABLE ")" / FONCTION_LISP*
PREDICAT	:=	*nature / genre / personne / ...*
VALEUR	:=	*NOMBRE / VARIABLE / nom / adj /...*
VARIABLE	:=	*CHAINE / "?"VAR*
VAR	:=	*a/b/c/d/e/ ...*
CLAUSE_ET_SI	:=	*"(" "ET_SI" LISTE_CONDITIONS ")"*

exemples :

	conditions	*commentaires*
(SI	*(unif mot (conc ?x "al")))*	*si* mot *se termine par* "-al"
(ET_SI	*(nature nom mot)*	*et si il existe comme nom dans la langue*
	(nombre sing mot)	*au singulier*
	(genre ?g mot))	*de genre ?g*

(SI (appartient mot L_PLUR_AL)) *(ET_SI)*	*si* mot *appartient à l'ensemble L_PLUR_AL*

avec :
L_PLUR_AL = ("bal", "cal", "cantal", "carnaval", "chacal", "festival", "régal")

c) Les conclusions

Expression d'un conclusion

La *Clause_Alors* est une suite ordonnée de conclusions exprimées dans le langage décrit ci-dessous.

Vocabulaire Le vocabulaire utilisé dans l'expression des conclusions est le même que celui employé dans l'expression des conditions.

Syntaxe Les termes qui apparaissent dans les conclusions sont des *termes-actions* de la forme $(A_i\ a_1 ... a_n)$ où les A_i sont des noms de procédures externes (fonctions Lisp) et les a_j des termes quelconques.

exemples :

Voici des exemples de fonctions dont nous donnons la signification dans le paragraphe suivant.

(egal ?y (conc ?x "aux"))

(ajouter_hyp ((nature nom)(nombre plur)) ?x)

Sémantique Tout *terme-action* est interprété par rapport aux informations spécifiées par le contexte de validation de la règle, obtenu après la vérification des conditions, c'est à dire à partir des hypothèses de la base de travail sélectionnées dans ce contexte de validation.

Deux fonctions sont principalement utilisées dans les règles qui ont servi à notre expérimentation : la fonction *egal* et la fonction *ajouter_hyp*. La fonction *(egal var val)* permet de donner à la variable non instanciée *var* la valeur *val* .

La fonction *(ajouter_hyp hyp var)* modifie la base de travail en ajoutant la nouvelle hypothèse grammaticale *hyp* (constituée d'une suite de couples prédicat-valeur) sur l'objet

qui instancie *var*. Nous rappelons que les hypothèses grammaticales sur un mot, stockées dans la base de travail, décrivent un ensemble d'usages de ce mot autorisés dans la langue. Elles ne constituent qu'un ensemble de possibilités dans lequel les analyses futures devront piocher pour déterminer celle qui est effective pour le mot employé dans une phrase.

La clause ALORS

La *Clause_Alors* d'une règle est exprimée à l'aide de la grammaire suivante.

CLAUSE_ALORS	:=	*"(" "ALORS" LISTE_CONCLUSIONS ")"*
LISTE_CONCLUSIONS	:=	*CONCLUSION**
CONCLUSION	:=	*FONCTION_LISP*

exemple

conclusions	*commentaires*
(ALORS (egal ?y (conc ?x "-aux"))	*alors construire la nouvelle variable ?y formé du radical ?x et de la désinence* "-aux"
(ajouter_hyp ((nature nom)	*et ajouter à la base de travail*
(nombre plur)	*l'hypothèse que ?y peut être un nom*
(genre ?g))	*au pluriel*
?y))	*et de genre ?g*

d) Le réseau des règles

Les *Clause_Ou* et *Clause_Sauf* pointent respectivement vers une liste de règles qui expriment un comportement différent dans une même situation et vers une liste de règles qui traduisent les exceptions.

Vocabulaire Le vocabulaire qui permet de décrire ces renvois est formé de numéros de règles.

Syntaxe Les termes apparaissant dans les *Clause_Sauf* et *Clause_Ou* sont des listes de numéros.

CLAUSE_SAUF	:=	*"(" "SAUF" LISTE_NUMERO ")"*
LISTE_NUMERO	:=	*NUMERO **
NUMERO	:=	*CHIFFRE **
CLAUSE_OU	:=	*"(" "OU" LISTE_NUMERO ")"*

3.3.3 Implantation

Une règle est définie comme une structure. Chacun de ses champs qui correspond à une clause est une fonction exprimée dans le langage Le_Lisp [Chailloux 86]. Les arguments de ces fonctions sont des listes de conditions à vérifier, d'actions à accomplir ou de renvois à d'autres règles.

Chaque *terme-relation* et chaque *terme-fonction* est une fonction Lisp. Cela permet d'interpréter chacun de ces termes par un simple appel de fonction.

3.3.4 Exemple de règles

règle			*commentaires*
(107	*init_morpho*		
(SI	*(unif mot (conc ?x "et")))*		*si* mot *se termine par* "et"
(ET_SI	*(nature adj mot)*		*et si il existe comme adjectif*
	(genre masc mot)		*au masculin*
	(nombre sing mot))		*et au singulier*
(ALORS	*(egal ?y (conc ?x "ette"))*		*alors construire la variable ?y en ajoutant* "ette" *au radical ?x*
	(ajouter_hyp	*((nature adj)*	*et ajouter à la base de travail*
		(genre fem)	*l'hypothèse que ?y peut être un adjectif*
		(nombre sing))	*au féminin*
		?y))	*et au singulier*
(SAUF	*(118))*		*sauf si la règle 118 s'applique sur mot*
(OU))			

(118 init_morpho			
(SI	*(appartient mot L_ET))*		*si* mot *appartient à l'ensemble L_ET*
	(unif mot (conc ?x "et"))		*et qu'il se construit ?x*"et"
(ET_SI)			
(ALORS	*(egal ?y (conc ?x "ète"))*		*alors construire ?y en ajoutant* "ète" *à ?x*
	(ajouter_hyp	*((nature adj)*	*et ajouter dans la base de travail*
		(genre fem)	*l'hypothèse que ?y peut être un adjectif*
		(nombre sing)	*au féminin*
		?y))	*et au singulier*
(SAUF)			
(OU))			

avec : L_ET = ("complet" "concret" "désuet" "discret" "incomplet" "indiscret" "inquiet" "replet" "secret")

* * *

Nous présenterons dans les chapitres suivants la manière d'utiliser la représentation adoptée en fonction des différentes tâches pour lesquelles doivent être utilisées les règles morphologiques et comment en exploiter les propriétés pour mettre en œuvre des raisonnements de finesse variable.

Pour l'analyse d'une phrase, le mot est considéré dans son contexte syntaxique. Les informations apportées par les règles morphologiques et le lexique sur l'usage des mots dans la langue sont propagées aux mots de la phrase par l'intermédiaire des stratégies. A partir de ces hypothèses, l'exploitation des règles syntaxiques, décrites dans le paragraphe suivant, permet de déterminer la structure de la phrase et les anomalies qu'elle présente. Le formalisme choisi pour les connaissances syntaxiques étant similaire à celui des règles morphologiques, nous avons préféré repousser pour une présentation unique l'exposé des mécanismes d'exploitation des règles (chapitre 4 et chapitre 5).

3.4 Les connaissances syntaxiques

Les relations entre les constituants d'une phrase en langue naturelle sont nombreuses et les contraintes qui les régissent sont multiples : contraintes d'ordonnancement, contraintes d'accord, présence nécessaire de complément ... Un *syntagme* est une *suite grammaticale*,

ou suite de mots reconnue par la syntaxe de la langue. Les connaissances syntaxiques ont pour rôle essentiel la détection des syntagmes dans lesquels les accords ne sont pas respectés.

Nous ne décrirons pas ici les modèles grammaticaux utilisés de façon régulière dans le traitement automatique du langage naturel. On peut se référer pour cela à [Sabah 88]. Il est communément admis que les grammaires hors-contexte au sens de Chomsky permettent de décrire un grand nombre de phénomènes de la langue. Le mécanisme de réécriture décrit par Chomsky peut être complété par un certain nombre de contraintes et permettre ainsi de tenir compte de phénomènes tels que les règles d'accord. Le modèle choisi est apparenté à ce type de grammaire.

3.4.1 Structure des règles syntaxiques

exemples de connaissances syntaxiques [Dubois 73] :

1. ∇ *Un groupe nominal étendu peut être formé d'un déterminant suivi d'un adjectif et d'un nom.*
2. ∇ *Un adjectif qualificatif épithète s'accorde en genre et en nombre avec le nom auquel il se rapporte.*
3. ∇ *Quand l'adjectif qualificatif est un constituant du groupe du nom, il a la fonction d'épithète du nom.*

Ces lois expriment indifféremment la description de constructions syntaxiques autorisées en français (*exemple 1*), les fonctions remplies par les différents constituants d'un syntagme (*exemple 3*), les contraintes d'accord apparaissant entre ses éléments (*exemple 2*). Ces informations peuvent être synthétisées en une seule règle, appelée *règle syntaxique*, qui définit en parallèle une structure syntaxique et les contraintes que ses constituants doivent vérifier.

exemple :

la règle suivante reprend les lois exprimées dans les exemples *1*, *2* et *3* :

∇ *Si un déterminant, un adjectif et un nom sont contigus dans la phrase, ils peuvent former un groupe nominal étendu. Ce déterminant et cet adjectif s'accordent alors en genre et en nombre avec ce nom.*

Une *règle syntaxique* exprime :

- un environnement d'application décrivant principalement une combinaison de différentes unités syntaxiques.

- des contraintes d'accord ;

- la catégorie de la nouvelle unité syntaxique que peuvent former les constituants syntaxiques sélectionnés ;

- elle est complétée par les valeurs que prennent les attributs grammaticaux de ce syntagme.

Une règle syntaxique est une règle locale dont la conclusion prise de façon isolée n'a qu'une valeur d'hypothèse. Cette conclusion n'est réellement validée que lorsqu'une structure de la phrase qui l'intègre est reconnue.

exemple :

Dans l'énoncé :

"chaque objet contrôle ses variables"

il est possible que "contrôle", *par application de la règle 516*[4], *soit considéré comme un groupe nominal simple, tandis que l'application de la règle 537 page 210 amène à supposer que* "contrôle" *est le prédicat de la phrase.*

Une structure analogue à celle des règles morphologiques ressort de ces informations :

- Le premier ensemble de conditions forme un filtre (dans les notations sous forme de règles de réécriture, cela correspond à la partie droite d'une règle).

- Un second ensemble des conditions précise ce contexte d'application : attributs grammaticaux des constituants, accords.

  L'application d'une règle peut être envisagée malgré une validation incomplète. La vérification des accords qui peuvent être sources d'erreurs peut par exemple être repoussée dans une étape ultérieure, soit pour approfondir les résultats, soit pour signaler les fautes commises sur la construction de la phrase (cf §5.3.2). Il faut, pour autoriser ces approximations, marquer les conditions dont la vérification peut être retardée et qui représentent en fait les contraintes de la langue sur lesquelles l'utilisateur risque de faire des erreurs que nous souhaitons traiter[5].

4. (cf page 205) L'ensemble des règles citées dans les exemples est donné en annexe 3.

5. Nous n'avons pas ressenti ce besoin quand nous avons fixé la représentation des règles morphologiques. Il n'est cependant pas exclu que, au cours de leur mise au point, certaines d'entre elles intègrent un ensemble de telles conditions.

- Les règles morphologiques sont organisées en un réseau hiérarchique. Cela permet de décrire des situations qui s'affinent progressivement. Certains phénomènes des règles syntaxiques présentent aussi une telle organisation.

  exemples :

  ∇ *Dans une proposition formée d'un groupe nominal gn1 suivi d'un groupe verbal et d'un groupe nominal gn2, le groupe nominal gn1 s'accorde en nombre et en personne avec le groupe du verbe.*

  ∇ *Mais si le verbe de cette proposition est un verbe copule (être, sembler, devenir ...), le groupe nominal gn2 est alors attribut du sujet et doit s'accorder en genre et en nombre avec lui.*

  ∇ *Un verbe peut former à lui tout seul un groupe verbal.*

  ∇ *Mais si le verbe est précédé d'une particule préverbale, alors le groupe verbal est formé de la particule et du verbe.*

3.4.2 Formalisme de description des règles syntaxiques

Le formalisme des règles syntaxiques reprend celui des règles morphologiques et intègre un niveau de conditions supplémentaire. Cet ensemble est désigné par la Clause *Sous_Réserve*. Il est constitué d'une liste de conditions définies dans le même langage que celui des Clauses *Si* et *Et_Si* et réunit les conditions dont la vérification peut être repoussée (cf règle 560 page 66). Par ce biais diverses stratégies pourront être développées, favorisant par exemple dans un premier temps les phrases correctes par rapport aux phrases erronées mais autorisant ensuite une analyse plus tolérante en cas de blocage dû à des contraintes d'accords non respectées.

La figure 3.4 qui complète la figure 3.3 donne la structure générale des règles syntaxiques.

Les langages utilisés pour exprimer conditions et conclusions sont les mêmes que ceux décrits par les règles morphologiques. Par contre les règles syntaxiques opèrent sur des *objets constituants* de la phrase, d'où la distinction entre le *mot de la phrase* et le *mot de la langue* (cf page 51).

SI	*"conditions d'activation de la règle"*
ET_SI	*"conditions précisant le contexte d'application"*
SOUS_RESERVE	*"conditions de validité qui peuvent être vérifiées ultérieurement"*
ALORS	*"conclusions admises sous réserve d'avoir effectuer toutes les vérifications précédentes et qu'aucune règle exception ne puisse s'appliquer"*
SAUF	*"liste des règles contredisant les conclusions pour les différentes exceptions"*
OU	*"liste des règles pouvant être appliquées pour le même phénomène"*

Figure 3.4. *forme générale des règles de syntaxe*

exemple :

Dans l'énoncé :

"Un segment parasite est inclus dans un groupe de segments parallèles."

on différencie l'objet *"un"* du lexique et les objets *"un"* premier mot et *"un"* 7ème mot de la phrase.

Un objet de la phrase est représenté par la chaîne de caractères qui le compose et par sa place dans la phrase, c'est à dire la place du premier mot dont il est formé.

exemple :

("dans un groupe" 6) représente le groupe de mots "dans un groupe" qui commence à la sixième place dans la phrase.

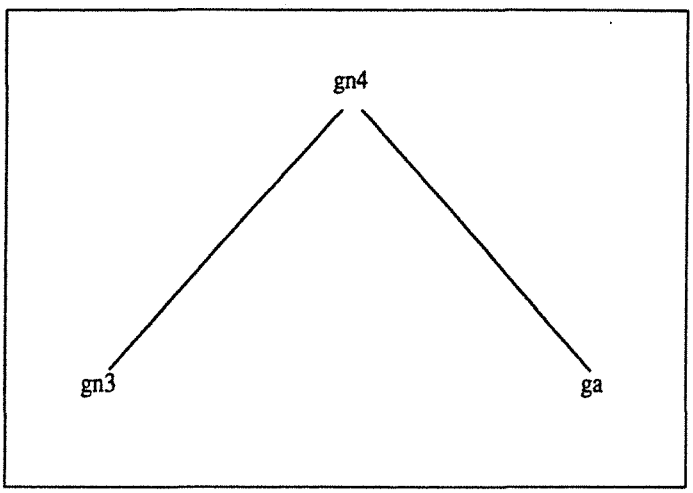


Figure 3.5.

	règle		*commentaires*
(513 syntaxique			
(SI	*(nature gn3 ?x)*		*si ?x peut être un gn3*
	(nature relative ?y)		*suivi de ?y qui peut être une relative*
	(succ ?x ?y))		
(ET_SI	*(nombre ?n ?x)*		*si ?x est de nombre ?n*
	(genre ?g ?x))		*de genre ?g*
(SOUS_RESERVE	*(participe ?z ?y)*		*sous réserve que si ?y admet un participe ?z*
	(nombre ?g ?z))		*il ait même nombre*
	(genre ?g ?z))		*et même genre que ?x*
(ALORS	*(egal ?w (conc ?x ?y))*		*alors construire ?w*
	(ajouter_hyp	*((nature gn4)*	*et ajouter l'hypothèse que ?w*
		(nombre ?n)	*peut être un gn4*
		(genre ?g)	*de même nombre,*
		?w))	*et même genre que ?x*
(SAUF)			
(OU))			

Sur l'énoncé suivant, nous allons illustrer la sémantique du lien OU dans les règles syntaxiques.

"le modèle contient des frontières d'îles représentées de la façon suivante."

On suppose qu'à une étape donnée, *("façon" 11)* a été reconnu comme un gn3 et *("suivante" 12)* comme un groupe adjectival. La règle 560 peut être appliquée et donner naissance à la structure syntaxique 3.6 :

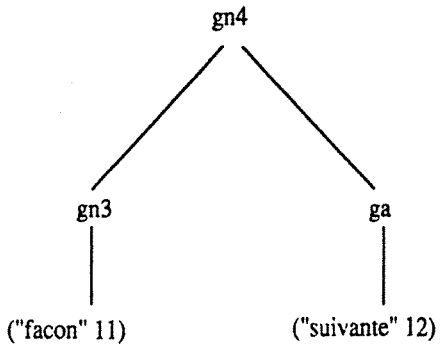


Figure 3.6.

Dans le groupe de mots *("frontières d'îles représentées" 5)*, on ne sait pas si le groupe adjectival *("représentées" 8)* se rapporte au groupe nominal simple (gns) *"îles"* ou au groupe nominal simple *("frontières" 5)*. Il y a ambiguïté sur l'application des règles 509 et 560 filtrées pour *(?x ("îles" 7)* et qui donnent respectivement naissance aux structures syntaxiques décrites par les arbres (1) et (2). C'est pourquoi elles sont liées par un *OU*.

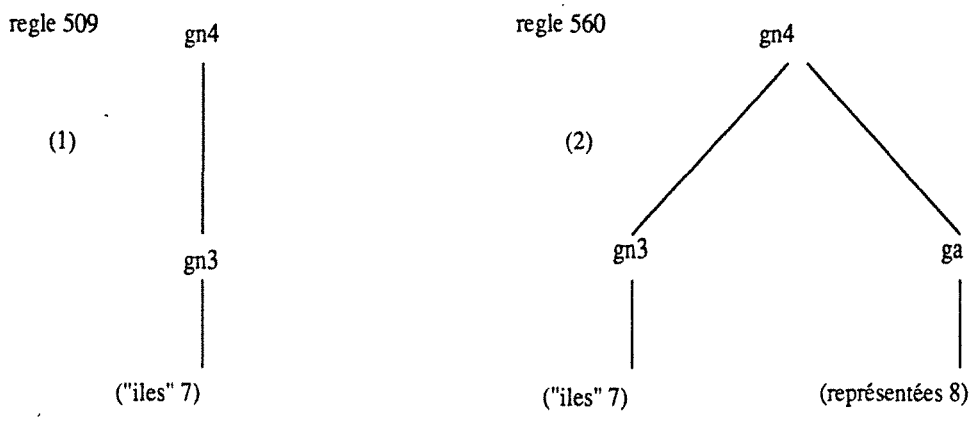


Figure 3.7.

Exploiter ces liens, pour favoriser l'une ou l'autre des règles, pour extraire de l'information pour la détection de fautes ..., relève plus que d'un simple mécanisme d'exploitation et nous avons préféré aborder ce problème dans la partie traitant des stratégies.

* * *

Il s'agit maintenant, après avoir défini le formalisme de représentation des règles du français, de fournir les moyens de les utiliser. Les outils mis à disposition de l'utilisateur et le mode d'expression des stratégies sont décrits dans le chapitre 4.

4

Exploitation des connaissances Expression des stratégies

Le chapitre précédent présentait le formalisme de description des connaissances, sans expliquer comment fixer l'usage qu'on veut en faire. Il est possible d'envisager un mode d'utilisation des règles qui, même s'il nécessite une réorganisation de nature compilatoire de ces dernières, reste simple et classique. Ce n'est plus le cas lorsqu'on veut tenir compte de l'aspect stratégique de la connaissance d'un expert et que l'on désire préciser les raisonnements à développer, jouer sur le degré de finesse des opérations, faire coopérer des sources diverses d'informations...

Le but de ce chapitre est d'exposer les différentes utilisations des règles et de présenter le mode d'expression proposé pour les stratégies. En effet, nous n'avons pas eu envie de figer le système une fois pour toutes avec un programme connaissant a priori la démarche à suivre et les actions à entreprendre, c'est pourquoi les différents éléments de stratégie sont exprimés de façon atomique et modulable dans des *règles de contrôle.*

Nous n'avons pas l'intention dans ce chapitre de présenter un développement complet de stratégies, mais simplement d'examiner la palette des moyens offerts pour les décrire. Les expérimentations présentées dans le chapitre 5 illustrent les idées exposées ici et donnent un aperçu sur la façon de définir les différentes stratégies sur l'exploitation des règles du français et de les transcrire dans des règles de contrôle.

4.1 Inférences sur les règles du français

4.1.1 Introduction

Les règles du français sont conçues, ou tout au moins formulées, pour résoudre un problème donné : énoncer la façon correcte de construire tout ou partie d'une phrase

et d'orthographier les mots la constituant. L'objectif général que nous poursuivons ici est de repérer le mauvais usage de ces règles, source des fautes d'orthographe auxquelles nous nous intéressons. Cet objectif global peut être décomposé en objectifs partiels ne correspondant pas forcément aux problèmes pour lesquels les règles ont été écrites. Ceci explique la nécessité de préciser comment les utiliser en indiquant :

- les prétraitements de nature compilatoire à leur faire subir (cf 4.1.2) ;
- la façon de les filtrer, de les valider, de les confirmer et de les entériner pour enfin les appliquer.

Précisons la signification de ces opérations introduites pour faciliter la compréhension de ce qui suit :

Filtrer une règle : Vérifier les conditions de la Clause *Si* d'une règle.
Valider une règle : Vérifier les conditions de la Clause *Et_Si* d'une règle.
Confirmer une règle : Vérifier les conditions de la Clause *Sous_Réserve* d'une règle.
Entériner une règle : Vérifier que les règles d'exception concernant une règle ne sont pas applicables.
Appliquer une règle : Exécuter les fonctions contenues dans la clause *Alors* d'une règle.

Une règle est déclarée **applicable** lorsque ses conditions d'application et ses contraintes d'exceptions ont été vérifiées modulo les options choisies pour l'analyse courante[1].

Revenons à la question posée et mentionnons les problèmes que l'on peut directement résoudre avec les règles du français et qui se trouvent exprimés dans la partie *Alors* des règles du français.

- Les règles morphologiques peuvent directement servir à résoudre deux problèmes, ce que nous allons illustrer avec la règle 50, règle générale sur la construction du pluriel des noms.

1. Un certain nombre d'approximations pourront en effet être autorisées par les stratégies (cf §5.1.3).

règle		*commentaires*
(50	*init_morpho*	
(SI	*(unif mot ?x))*	*si on unifie ?x et mot*
(ET_SI	*(nature nom mot)*	*et si mot peut être un nom*
	(nombre sing mot)	*au singulier*
	(genre ?g mot))	*de genre ?g*
(ALORS	*(egal ?y (conc ?x "s"))*	*alors construire ?y en ajoutant "s" à ?x*
	(ajouter_hyp ((nature nom)	*et ajouter à la base d'hypothèses*
	(nombre plur)	*que ?y peut être un nom*
	(genre ?g))	*au pluriel*
	?y))	*de genre ?g*
(SAUF	*(51 56 52 53 57 59))*	*sauf si une des règles 51, 56, 52, 53, 57 ou 59 s'applique sur mot*
(OU	*(58)))*	*ou si on applique la règle 58*
(51	*init_morpho*	
(SI	*(unif mot (conc ?x "au")))*	*si mot s'écrit ?x "au"*
(ET_SI	*(nature nom mot)*	*et si mot peut être un nom*
	(nombre sing mot)	*au singulier*
	(genre ?g mot))	*de genre ?g*
(ALORS	*(egal ?y (conc ?x "aux"))*	*alors construire ?y en ajoutant "aux" à ?x*
	(ajouter_hyp ((nature nom)	*et ajouter à la base d'hypothèses*
	(nombre plur)	*que ?y peut être un nom*
	(genre ?g))	*au pluriel*
	?y))	*de genre ?g*
(SAUF	*(60)))*	*sauf si la règle 60 s'applique sur mot*

Les problèmes que peut résoudre la règle 50 sont :

Problème 1 : Construire la forme que prend un mot du lexique mis au pluriel.

Appliquée sur le mot *"bateau"*, cette règle proposerait la construction de *"bateaus"* mais s'efface en fait devant la règle d'exception 51 sur la construction du pluriel des noms terminés par *"au"* qui, elle, donne la forme correcte : *"bateaux"*.

Problème 2 : Etant donnés un mot du lexique *?x* et une forme fléchie *?y* sous contrainte d'accord grammatical, trouver cette contrainte.

- Appliquée pour les mots : *"description"* et *"descriptions"*,
  et l'hypothèse du lexique :
  ("description" (1 (nature nom) (nombre sing) (genre fem)))
  la règle 50 ajoute dans la base de travail :
  ("descriptions" (1 (nature nom) (nombre plur) (genre fem))).

- Appliquée pour les mots : *"bateau"* et *"bateaux"*,
  et l'hypothèse du lexique :
  ("bateau" (1 (nature nom) (nombre sing) (genre masc)))
  la règle 51 via la règle 50 donne :
  ("bateaux" (1 (nature nom) (nombre plur) (genre masc))).

- Les règles syntaxiques permettent de résoudre le problème suivant :

Problème 3 : Délimiter des constituants syntaxiques élémentaires dans une phrase.

La structure hiérarchique des règles syntaxiques permet de créer les syntagmes procurant un recouvrement maximum du texte. En effet, l'application d'une règle accessible par un lien *Sauf* qui interdit celle de la règle d'entrée, permet de créer un syntagme plus grand en intégrant celui qui aurait été créé par cette dernière.

	règle	*commentaires*
(516	*syntaxique*	
(SI	*(nature nom ?x))*	*si ?x peut être un nom*
(ET_SI	*(nombre ?n ?x)*	*de nombre ?n*
	(genre ?g ?x))	*et de genre ?g*
(SOUS_RESERVE)		
(ALORS	*(ajouter_hyp((nature gns)*	*alors ajouter l'hypothèse*
	(nombre ?n)	*que ?x peut être un gns*
	(genre ?g))	*de nombre ?n*
	?x))	*et de genre ?g*
(SAUF	*(517 561))*	*sauf si une des règles 517 ou 561 s'applique*
(OU))		
(517	*syntaxique*	
(SI	*(nature nom ?x)*	*si ?x peut être un nom*
	(nature ga ?y)	*précédé d'un groupe adjectival*
	(succ ?y ?x))	
(ET_SI	*(nombre ?n ?x)*	*et si ?x est de nombre ?n*
	(genre ?g ?x))	*et de genre ?g*
(SOUS_RESERVE	*(nombre ?n ?y)*	*sous réserve que ?y ait même nombre*
	(genre ?g ?y))	*et même genre*
(ALORS	*(egal ?z (conc ?y ?x))*	*alors construire ?z avec ?y suivi de ?x*
	(ajouter_hyp((nature gns)	*et ajouter l'hypothèse*
	(nombre ?n)	*que ?z peut être un gns*
	(genre ?g))	*de nombre ?n*
	?z))	*de genre ?g*
(OU)		
(SAUF))		

Appliquée pour les constituants *("petit" 2)* et *("bateau" 3)*,
extraits de la phrase *"le petit bateau coule"*,

et les hypothèses[2]de la base de travail[3] :

(("petit" 2) (3 (nature adj)(genre masc)(nombre sing)))
et *(("bateau" 3) (1 (nature nom)(genre masc)(nombre sing)))*

La règle 516 donnerait :

(("bateau" 3) (1 (nature gns)(nombre sing)(genre masc)))
mais doit s'effacer devant la règle 517 qui donne :

(("petit bateau" 2) (1 (nature gns)(nombre sing)(genre masc)))
qui offre un meilleur recouvrement du texte.

Le lien *Ou* permet aussi de travailler dans ce sens en offrant la possibilité de privilégier, lorsqu'il y a ambiguïté sur l'application d'une règle ou d'une de ses règles *Ou*, celle qui offre un meilleur recouvrement du texte, quitte à revenir sur cette décision en cas d'échec (cf §5.3.3).

4.1.2 Textes sans fautes

a) D'une règle à ses objectifs

Comme nous allons l'illustrer par un exemple, deux types d'objectifs sont à satisfaire pour pouvoir déclarer sans faute un énoncé :

Objectif 1 : Etant donné un mot, trouver l'hypothèse grammaticale qui correspond à sa forme ;

Objectif 2 : Délimiter une entité syntaxique au sein d'une séquence de constituants plus élémentaires.

Ces objectifs correspondent aux problèmes que savent résoudre les règles.

---

2. Nous rappelons que :
- le numéro qui suit le(la liste de) mot(s) correspond au rang du (premier) mot (de la liste) dans la phrase ;
- le numéro en tête des analyses proposées correspond au numéro de l'hypothèse développée.

3. Pour des raisons de clarté, nous n'avons fourni ici que les hypothèses grammaticales pertinentes pour l'analyse effectuée.

En effet, constater que la phrase :

"le petit bateau coule"

est bien construite et bien orthographiée revient à satisfaire l'objectif de type 2 :

(nature prop *("le petit bateau coule" 1*))

en obtenant comme résultat d'analyse l'hypothèse suivante dans la base de travail :

((*"le petit bateau coule" 1*) (1 (nature prop) ...))

Ce résultat peut s'obtenir en exploitant les règles du français en chaînage avant ou en chaînage arrière, comme nous l'avons schématisé par la figure 4.1.

Il est à remarquer toutefois que :

- la hiérarchie des règles (liens *Sauf* et *Ou*) oblige, de façon à limiter les points d'entrée et à repérer les cas d'exception, à un même ordre de prise en considération des règles, règles générales puis règles particulières, quel que soit le sens du chaînage. Les règles à consulter en priorité sont les *règles d'entrée* qui conviennent à la situation, c'est à dire celles qui sont les plus générales pour traiter une question spécifique (par exemple la règle 51 (page 73) est la règle d'entrée du pluriel des noms terminés par *"au"*, la règle 516 (page 205) est la règle d'entrée pour la construction de groupes nominaux simples (*gns*)) ;
- l'absence d'un mot de la phrase tel quel dans le lexique (par exemple *"coule"*) contraint à employer au moins localement le chaînage arrière (symbolisé par des pointillés sur la figure 4.1) sur les règles morphologiques. Il faut en effet, à partir des conclusions apportées par les règles, retrouver celles qui ont pu construire le mot et en vérifier les conditions d'application. Le paragraphe suivant présente la démarche suivie pour mettre en œuvre ce chaînage arrière.

b) Chaînage arrière et précompilation

Utilisé sans précaution, le chaînage arrière génère une combinatoire importante. Afin de remédier à cet inconvénient, on peut envisager de doubler les règles morphologiques par d'autres règles résolvant directement le problème posé ci-dessus. Ces règles auraient comme objectif nouveau de trouver des hypothèses grammaticales sur les mots à partir de la forme où ils sont écrits.

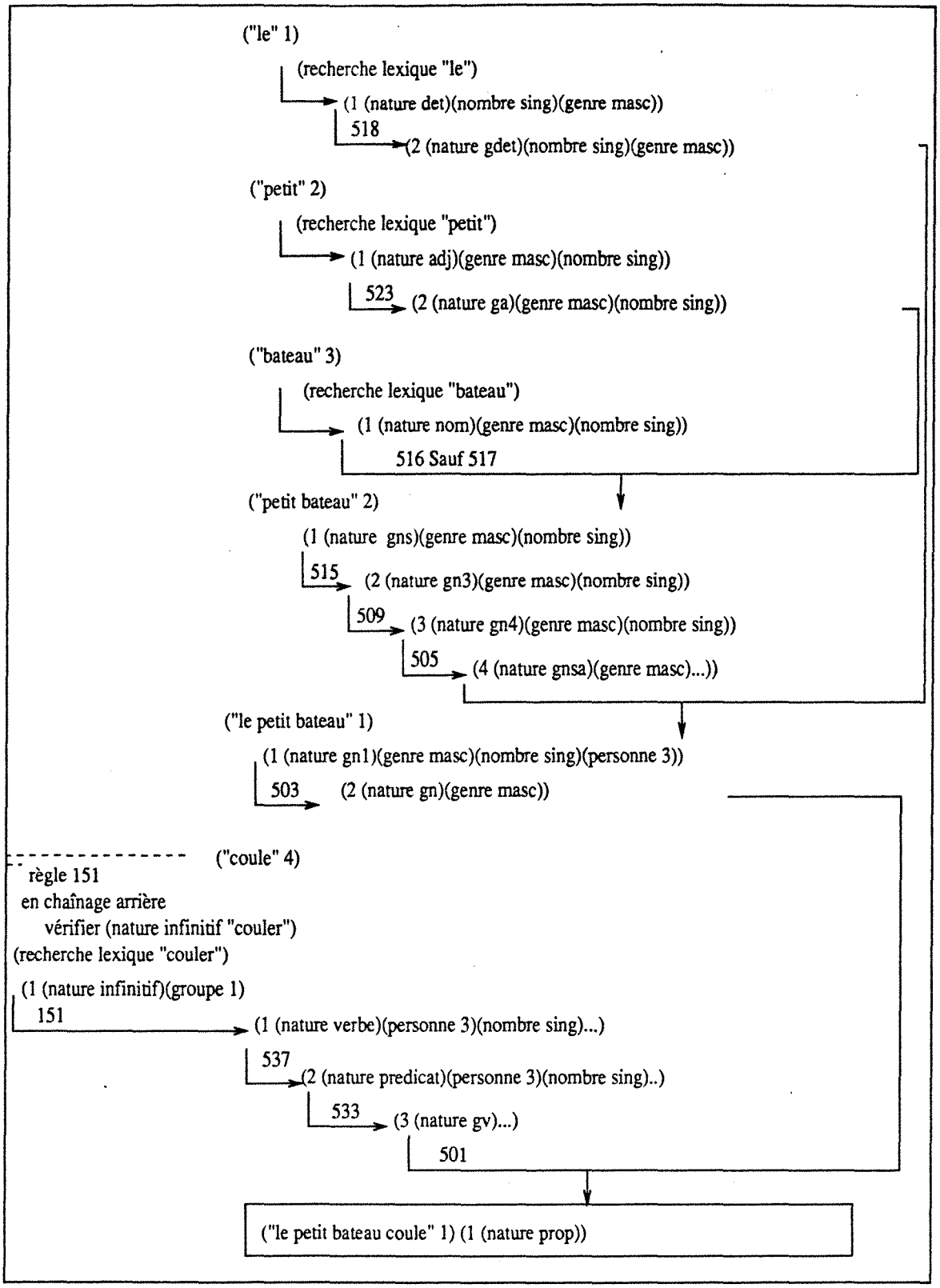


Figure 4.1. Analyse de la phrase *"le petit bateau coule"*

exemples :

règle		*commentaires*
(50bis	*morphologique*	
(SI	*(unif mot (conc ?x "s")))*	*si* mot *a la forme ?x*"s"
(ET_SI	*(nature nom ?x)*	*et si ?x peut être un nom*
	(nombre sing ?x)	*au singulier*
	(genre ?g ?x))	*de genre ?g*
(ALORS	*(ajouter_hyp ((nature nom)*	*alors ajouter l'hypothèse que mot*
	(genre ?g)	*peut être un nom*
	(nombre plur))	*de genre ?g*
	mot))	*au pluriel*
(OU	*(70bis 151bis ...)))*	*à moins qu'une des règles 70bis, 151bis,...ne s'applique*

règle		*commentaires*
(51bis	*morphologique*	
(SI	*(unif mot (conc ?x "aux")))*	*si* mot *se termine pas* "aux"
(ET_SI	*(egal ?y (conc ?x "au"))*	*et si ?y est égal à mot sans le* "x"
	(nature nom ?y)	*et qu'il peut être un nom*
	(nombre sing ?y)	*au singulier*
	(genre ?g ?y))	*de genre ?g*
(ALORS	*(ajouter_hyp ((nature nom)*	*alors ajouter l'hypothèse que mot*
	(genre ?g)	*peut être un nom*
	(nombre plur))	*de genre ?g*
	mot))	*au pluriel*
(OU	*(71bis)))*	*à moins que la règle 71bis ne s'applique*

Cela reviendrait à demander à l'*expert orthographe* de décrire les connaissances déjà fournies dans les règles morphologiques intiales sous une autre forme. La présence d'informations redondantes est toujours une source probable d'erreurs, et plus spécifiquement d'incohérences supplémentaires qu'il vaut mieux éviter dans la base de connaissances.

En effet, cette façon d'opérer nécessite de s'assurer de la cohérence des conditions entre les règles intiales et les règles *bis* et de revoir les hiérarchies *Sauf* et *Ou*.

De plus, en faisant abstraction des règles initiales, on perd l'intérêt que leur organisation hiérarchique et la structure progressive de leurs conditions offre pour les stratégies (cf 5.1.3, 5.2.2, 5.3.2).

C'est pourquoi nous pensons préférable d'utiliser un traitement de nature compilatoire qui permette de renvoyer aux règles initales pour retrouver les informations. Ce traitement consiste à définir une fois pour toutes, en fonction de la forme du mot, les objets sur lesquels vérifier les conditions des règles initiales de façon à déclencher ensuite ces dernières en chaînage avant.

Appliqué aux règles de morphologie, ce traitement fournit l'équivalent d'un automate des désinences (cf annexe 3) dont les sorties mentionnent les règles intiales les plus générales associées à chaque désinence, ainsi que les racines correspondantes. Cette compilation, effectuée par un chaînage arrière, a été pratiquée manuellement dans le cadre de notre expérimentation, nous en donnons ci-après des extraits intéressant l'analyse du mot *"coule"*.

Cette façon d'opérer pour la sélection des règles morphologiques peut être comparée, dans une certaine mesure, avec ce qui est appelé le paramètre *mis-à-jour* exploité dans le célèbre système expert MYCIN.

Le paramètre "mis-a-jour" dans MYCIN MYCIN est avec Dendral ou Prospector un des pionniers de la méthodologie des systèmes experts. Il opère dans le domaine du diagnostic médical.

Les règles de MYCIN comportent deux parties notées *Prémisse* et *Conclusion*. Le moteur de MYCIN fonctionne en chaînage arrière. C'est à dire que généralement une règle est invoquée parce qu'elle traite en partie *Conclusion*, d'une proposition qui, à l'instant courant, intéresse le moteur de MYCIN. Les conditions des parties *Prémisse* se réfèrent à des objets appelés *contexte*. A chaque instant, le moteur de MYCIN s'intéresse à un *contexte* particulier. Le moteur de MYCIN est conçu pour évaluer les *paramètres cliniques* associés au contexte courant.

Pour ce faire, MYCIN doit associer à chaque type de contexte un ensemble de caractéristiques appelés *paramètres cliniques*. A certains *paramètres cliniques*, est associée une propriété *"mis-a-jour"* dont la valeur est une liste des règles qui sont capables de conclure quant aux valeurs éventuelles du paramètre.

?x"E" *renvoi aux règles 1 et 26, 101 et 126, 130* *sur le radical* ?x *et à la règle 151* *sur le radical* ?x "ER"		
OU	?x"TRICE" *renvoi aux règles 24, 25, 124, 125* *sur le radical* ?x"TEUR"	
	?x"ALE" *renvoi aux règles 9, 109* *sur le radical* ?x"AL"	
	?x"LLE" *renvoi aux règles 2, 102* *sur le radical* ?x"L"	
	OU	?x"ELLE" *renvoi aux règles 4, 104* *sur le radical* ?x"EAU" *et à la règle 157* *sur le radical* ?x"ELER"
		?x"OLLE" *renvoi aux règles 5, 105* *sur le radical* ?x "OU"
	...	

Figure 4.2. Extrait des sorties de l'automate morphologique.

Cette propriété permet de façon similaire au système d'automate que nous avons introduit dans notre système de sélectionner une liste de règles à invoquer dans une étape ultérieure.

* * *

De cette façon, le nouvel objectif ne correspondant pas aux 3 problèmes résolus par les règles est satisfait par un prétraitement de nature compilatoire indiquant les règles sous leur forme initiale à utiliser et comment les utiliser.

4.1.3 Textes avec fautes

L'analyse des phrases se complique en présence de fautes. En effet :

- les chaînages arrière et avant peuvent de ce fait se trouver bloqués et il faudra faire des hypothèses sur la cause de blocage si l'on veut pouvoir poursuivre l'analyse et détecter en particulier une éventuelle autre erreur dans la phrase ;
- les causes d'une faute sont multiples et il convient pour faciliter la tâche de l'utilisateur, sinon de les lui suggérer toutes, de lui donner au moins suffisamment d'informations pour qu'il les retrouve.

Ces difficultés ne sont d'ailleurs pas inhérentes à l'orthographe d'une langue. On les rencontre en compilation en général sur des langages formels. Les stratégies proposées dans le chapitre 5 traiteront de ces différents problèmes au niveau global, en attendant nous présentons les utilisations des règles à envisager dans de telles circonstances.

exemple :

Soit donc le traitement de la nouvelle phrase :

"les petits bateaus coule"

L'analyse bloque sur la forme erronée *"bateaus"*, mais une analyse ascendante permet d'obtenir tout de même les informations portées sur la figure 4.3.

On peut envisager plusieurs cas de figure possibles pour expliquer l'erreur sur *"bateaus"* :

- le mot *"bateaus"* aurait dû s'écrire *"bateaux"* ; la faute viendrait d'un mauvais emploi des règles sur la construction du pluriel des noms et plus particulièrement de l'oubli de la règle exception sur le pluriel des noms en *"au"*[4] ;
- le mot *"bateaus"* est en trop dans cette phrase.

Du fait de ce premier blocage, la deuxième erreur sur l'accord entre le groupe nominal (*"les petits bateaus" 1*) et le verbe (*"coule" 4*) n'a pas été envisagée.

Pour sérier les différentes possibilités sur l'origine du blocage, nous admettons que la conception hiérarchique des règles et de leurs conditions d'application donne des informations pertinentes sur les causes d'erreurs les plus probables. C'est à dire que dans le cas présent nous admettrons que la faute se détecte au niveau de la première règle d'exception

4. Nous nous intéressons aux fautes grammaticales dues au mauvais emploi des règles du français, c'est pourquoi nous ne prendrons pas ici l'hypothèse qu'il s'agit d'une faute typographique.

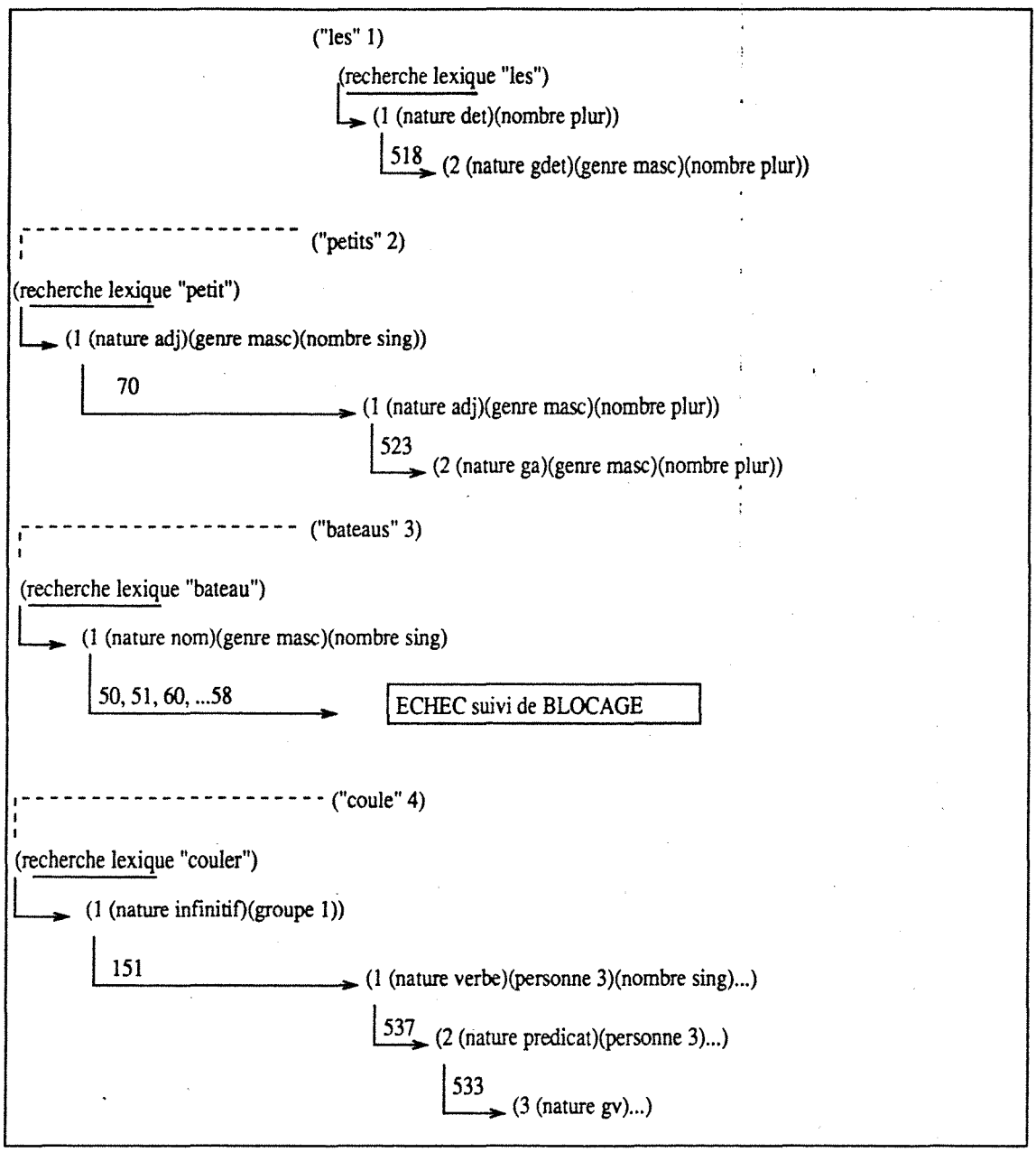


Figure 4.3. Analyse de la phrase *"les petits bateaus coule"*

défaillante. Pour l'exemple en cours, c'est la règle 50 (page 72) sur la construction du pluriel des noms qui est défaillante, sa règle d'exception sur le pluriel des noms terminés par *"au"* n'a pas été respectée.

La correction à apporter est alors indiquée par la(les) condition(s) non satisfaite(s). Ici cela correspond à l'application de la règle d'exception 51 (page 73) qui n'a pas été suivie et à la conclusion :

(egal ?y (conc "bate" *"aux"))*

On peut donc suggérer à l'utilisateur de remplacer *"bateaus"* par *"bateaux"*, et poursuivre l'analyse comme indiquée ci-après (figure 4.4), ou adopter une autre attitude suivant les stratégies adoptées basées par exemple sur la tolérance (cf 5.2.2).

	règle	*commentaires*
(501	*syntaxique*	
(SI	*(nature gv ?x)*	*si ?x peut être un gv*
	(nature gn ?y)	*précédé de ?y qui peut être un groupe nominal*
	(succ ?y ?x))	
(ET_SI	*(nombre ?n ?y)*	*si ?y est de nombre ?n,*
	(personne ?p ?y)	*de personne ?p*
	(participe ?w ?x))	*s'il admet un éventuel (facultatif) participe ?w*
(SOUS_RESERVE	*(nombre ?n ?x)*	*sous réserve que ?x ait*
	(personne ?p ?x))	*même nombre et même personne que ?y*
(ALORS	*(egal ?z (conc ?y ?x))*	*alors construire ?z*
	(ajouter_hyp ((nature prop)	*et ajouter l'hypothèse*
	(participe ?w))	*que ?z peut être une proposition*
	?z))	*avec le même participe ?w que ?x (s'il en avait un)*
(SAUF	*(502)))*	*sauf si la règle 502 s'applique*

La seconde faute bloque également l'analyse. Elle empêche l'application des règles adéquates pour construire la structure syntaxique de la phrase.

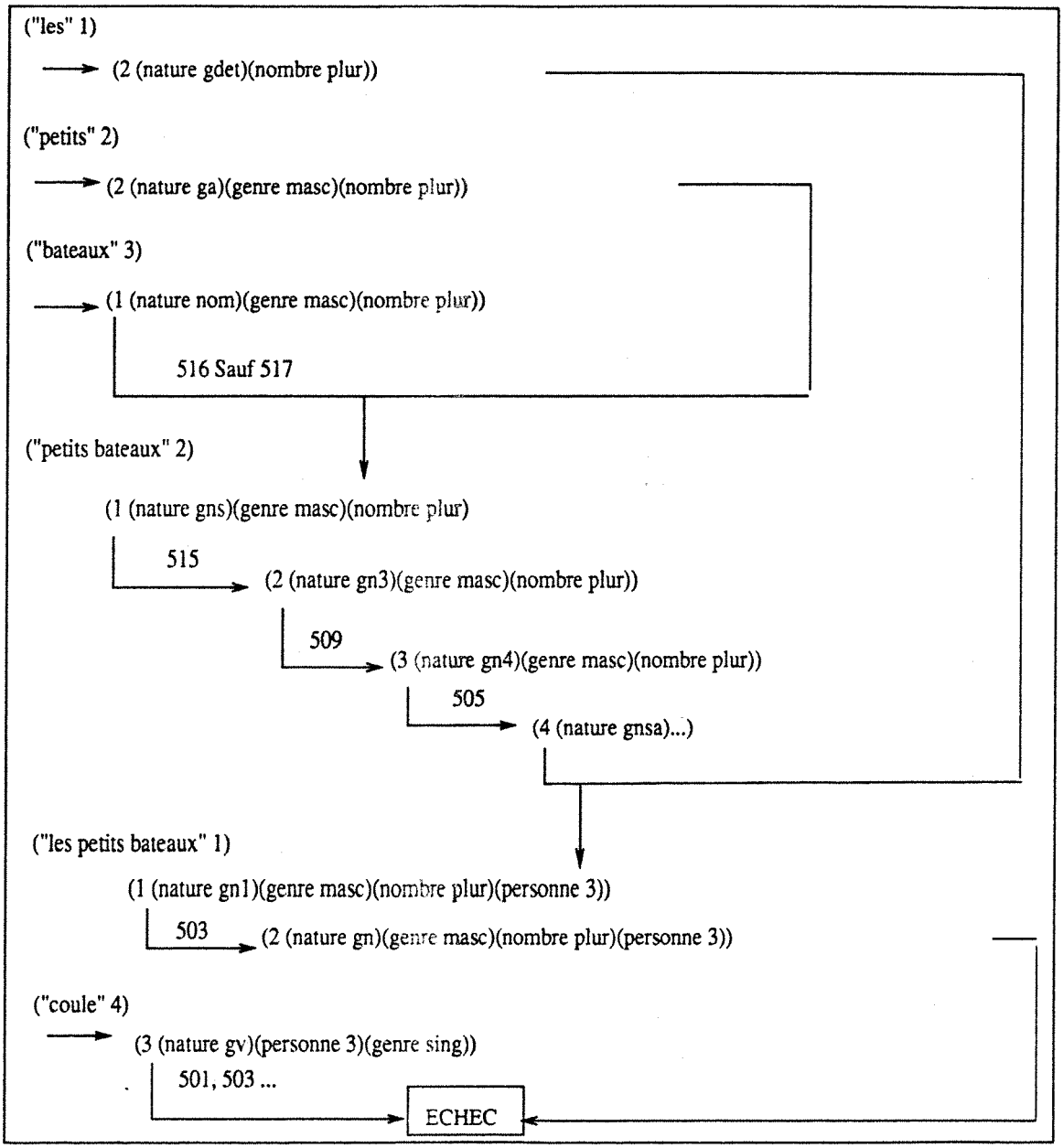


Figure 4.4. Poursuite de l'analyse de *"les petits bateaus coule"* après première correction.

La règle d'entrée 501 est la première de toutes les règles filtrées et validées qu'on n'arrive pas à satisfaire dans l'analyse en cours. Les conditions de la clause *Sous_Réserve* qui traduisent les conditions d'accord au sein du syntagme ne sont en effet pas toutes vérifiées.

Nous admettons que c'est sur la non application de cette règle que se trouve l'explication de l'erreur. La seule cause est la non satisfaction de la condition suivante de la clause *Sous_Réserve* :

(nombre plur "coule" *)*

A partir de ce diagnostic, on peut demander au rédacteur d'effectuer une correction en mettant ce verbe au pluriel. Il est même possible de proposer la solution à l'aide de la règle 165 (cf page 185) sur la conjugaison des verbes du premier groupe, aux personnes du pluriel du présent (problème de type 1).

La correction ainsi apportée permet de terminer l'analyse et de conclure sur la structure de la phrase (figure 4.5).

L'aboutissement de l'analyse à la suite de ces corrections offre un argument non négligeable quant à la validité de celles-ci.

Nous verrons au chapitre suivant des exemples moins triviaux, où en particulier il existe plusieurs possibilités d'interprétation de l'erreur du fait de l'existence de plusieurs règles non respectées. Nous verrons aussi comment inhiber dynamiquement les erreurs rencontrées et ne pas faire systématiquement appel à l'utilisateur pour effectuer les corrections proposées.

* * *

Cette présentation sur l'usage des règles et les compléments que nous apporterons au chapitre suivant, nous amènent à faire les réflexions suivantes concernant l'analyse d'une phrase et la détection des fautes qu'elle contient :

1. Nous pratiquons de préférence le chaînage avant ;
2. Il est intéressant de remplacer un chaînage arrière par la détection de points d'entrée possibles pour retrouver par chaînage avant les informations recherchées directement sur les règles initiales (cf 4.1.2) ;

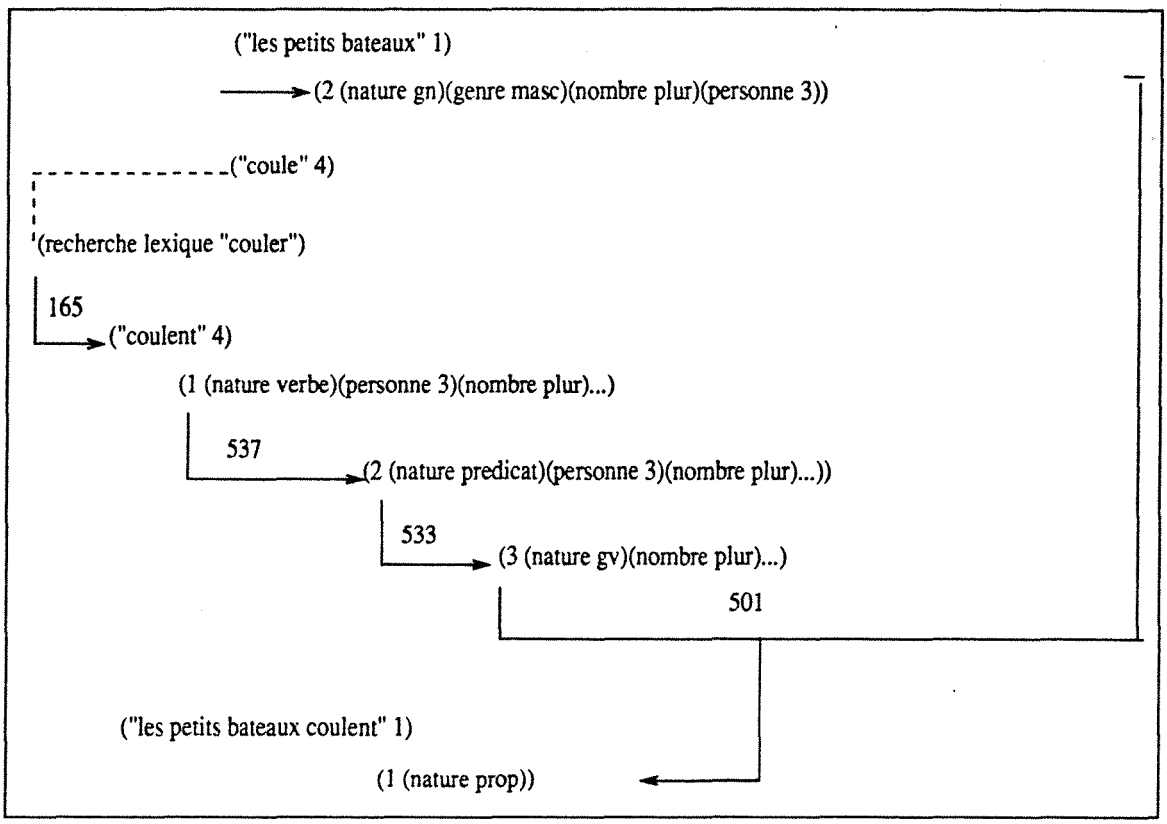


Figure 4.5. Poursuite de l'analyse de *"les petits bateaus coule"* après deuxième correction.

3. Il est intéressant, en cas d'échec, de ne pas tenir compte des règles exceptions (cf 5.2.2) ;

4. Il est intéressant, en cas d'échec également, de pouvoir continuer l'analyse en inhibant la vérification des conditions des clauses *Sous_Réserve* de certaines règles (cf 5.3.2) ;

5. Afin de réduire les hypothèses multiples et de ce fait la combinatoire, il est intéressant d'exploiter les liens *Ou* de façon à privilégier certaines hypothèses sur les structures ambiguës (cf 5.3.3).

La mise en œuvre de ces différentes alternatives se traduit par l'intermédiaire de paramètres modulant les fonctions de déclenchement des règles en chaînage avant et en chaînage arrière. Ils seront présentés plus en détail dans le chapitre suivant, au fur et à mesure des besoins des stratégies, mais nous pouvons tout de même citer pour mémoire les booléens `confirm` et `priorite_ou` exprimant respectivement les éléments de stratégie 4 et 5, et le

paramètre **prof** permettant de limiter la profondeur des règles consultées et de traduire le point 3.

4.2 Les règles de contrôle

Lorsque la détection des fautes est plus complexe, il devient nécessaire de définir des stratégies moins "opératoires". On peut reprendre le propos de [Pitrat 84] qui dit que de façon schématique la définition d'une stratégie d'analyse est liée à plusieurs expertises :

- Expertise pour exprimer un plan, c'est à dire un ensemble de séquences ou d'arborescences de sous-buts conduisant à l'objectif donné. Il faut prévoir un moyen de transformation du problème depuis une situation quelconque en une ou plusieurs autres situations intermédaires.

  La recherche de la structure syntaxique d'un groupe de mots, peut se décomposer par la reconnaissance de ses différents constituants, puis par l'étude de leur agencement dans la phrase.

- Expertise pour réaliser un objectif élémentaire. Quelles sont les méthodes possibles pour arriver à un certain type de résultat ?

  Les méthodes pour retrouver les hypothèses grammaticales sur un mot consistent :

  - à déclencher une action de recherche dans un lexique ;
  - à entreprendre une analyse morphologique.

- Expertise pour modifier un plan. Il est intéressant de revoir un enchaînement d'actions qui a échoué, de reprendre ou de compléter une étape du plan qui n'a pas donné les résultats escomptés. L'analyse des raisons de son échec peut aider à en construire un autre.

  La vérification de la clause *Sous_Réserve* des règles syntaxiques peut s'avérer trop sévère et même bloquante pour l'analyse syntaxique. Après un constat d'échec, il doit être possible de faire abstraction de certaines de ces conditions avant de reprendre les traitements.

Nous proposons, pour plus de souplesse et de modularité, d'exprimer ces expertises de façon déclarative dans des règles appelées *règles de contrôle.*

On distingue deux types de règles de contrôle :

- les *règles de stratégie* qui décrivent des plans ou des actions permettant de réaliser les différents objectifs.
- les *règles de reprise* qui sont utilisées en cas d'échec pour aider à contourner une stratégie qui a échoué.

4.2.1 Les règles de stratégie

Une *règle de stratégie* décrit une stratégie locale pour résoudre une catégorie de problèmes donnée. Cette stratégie se traduit sous forme d'*actions à entreprendre* ou de *problèmes plus simples* décomposant le problème initial.

exemples :

∇ *Pour connaître les catégories grammaticales d'un mot, on peut lance l'opération de recherche du mot dans le lexique puis on en effectue une analyse morphologique.*

∇ *Pour contrôler qu'une phrase ne contient pas de fautes d'orthographe, on peut :*

- *déterminer la structure syntaxique de la phrase,*
- *puis s'assurer qu'on n'a pas rencontré d'anomalies.*

a) Structure des règles de stratégie

La structure retenue pour les *règles de stratégie* permet de spécifier pour un objectif donné (partie *Si*) :

- sa décomposition en une séquence d'actions ou une séquence d'objectifs plus élémentaires (partie *Alors*) ;
- les conditions sous lesquelles cette décomposition peut se faire (partie *Et_Si*).

SI		*objectif à atteindre*
ET_SI		*conditions pour préciser le contexte d'application*
ALORS		*liste d'actions à entreprendre*
	ou	*sous-objectifs à rechercher*[5]

Figure 4.6. Structure des règles de contrôle

On retrouve une structure analogue à celle des règles sur la langue :

Le formalisme pour exprimer les conditions et les conclusions reprend celui des règles du français (chapitre 3). Seules les nouvelles conventions sont décrites dans les paragraphes suivants.

b) Expression et réalisation d'un objectif

Un objectif est soit un prédicat à vérifier, soit le marqueur d'une tâche à entreprendre. Ils sont tous de la forme (R_i a_1 ...a_n).

Si l'objectif représente un prédicat, on dit qu'il est *réalisé* (ou *atteint*) quand il existe dans la base de travail des hypothèses qui permettent d'instancier les variables qui le composent.

exemple :

L'objectif *(nature ?n "américaines")* est atteint avec les hypothèses suivantes de la base de travail :

("américaines" (1 (nature nom) (nombre plur)...)
et *("américaines" (2 (nature adj) (nombre plur)...).*

Quand la règle propose une décomposition du problème en problèmes plus élémentaires, chacun des sous-objectifs intermédiaires est introduit dans la clause *Alors* de la règle de stratégie par la fonction `res_avant`.

5. La différence entre les deux types de conclusions se fait comme on le verra plus tard par la fonction *res_avant* qui introduit les sous-objectifs à réaliser.

c) Exemple de règle de stratégie

(800	*stratégie*	
(Si	*(objectif (bien_ecrit ?n enonce)))*	*si l'objectif est de vérifier que l'énoncé est bien écrit*
(Alors	*(res_avant (bien_ecrit mots_isolés enonce))*	*alors chercher à vérifier que les mots pris isolément sont bien écrits*
	(res_avant (bien_ecrit ?n mots_en_contexte))))	*et chercher à vérifier que ces mots pris dans leur contexte sont également bien écrits*

L'exécution de la fonction `res_avant` a pour conséquence de lancer récursivement la recherche de l'objectif passé en paramètre. La réalisation successive de tous les sous-objectifs doit amener à la réalisation de l'objectif recherché.

4.2.2 Les règles de reprise

Les analyses pour réaliser un objectif donné sont déclenchées par l'intermédiaire des règles de stratégie. L'hypothèse d'un travail à profondeur variable fait naître l'ensemble des alternatives schématisées dans la figure 4.7.

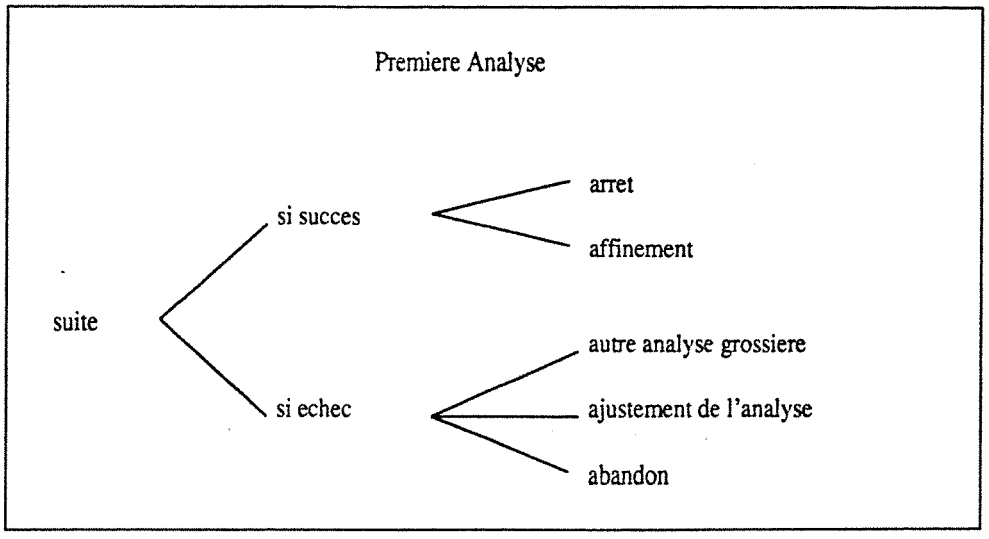


Figure 4.7.

Les règles de reprise interviennent en cas d'échec. Elles permettent de décrire comment adapter si c'est possible la stratégie suivie. Une règle de reprise est associée à une stratégie. En fonction de la situation rencontrée, des approximations qui ont été autorisées sur les traitements, elle propose des actions de récupération. Ces actions vont porter soit directement sur les opérations déclenchées dans les règles de stratégie, et notamment sur la précision des inférences effectuées, soit sur les sous-objectifs dont elle dépend. Une stratégie pour un objectif donné est donc définie par une règle de stratégie et une liste de règles de reprise qui seront prises en compte les unes après les autres dans l'ordre où elles sont fournies, de la façon dont nous l'avons représenté dans la figure 4.8.

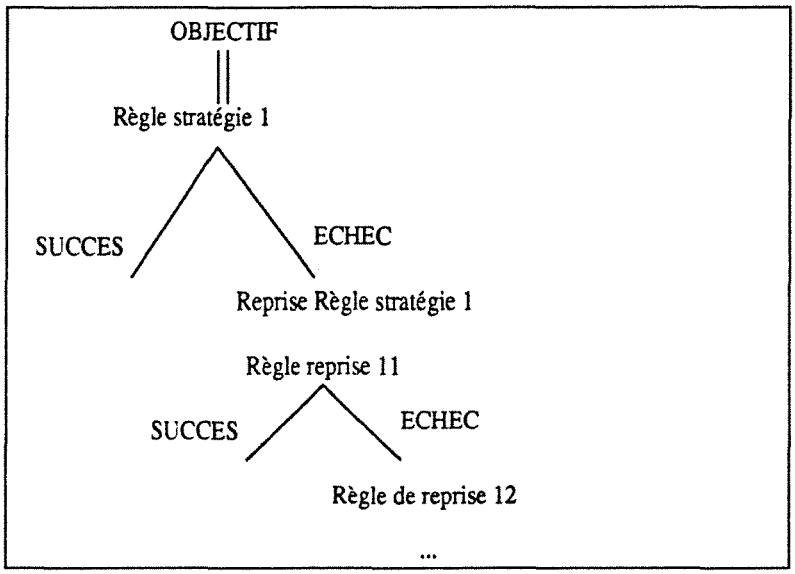


Figure 4.8. Prise en compte des règles de reprise.

Les règles de reprise sont exprimées dans le même formalisme que celui des règles de stratégie. Nous n'en donnerons pas d'exemple ici, il nous semble préférable de renvoyer pour cela au chapitre suivant où elles seront présentées au sein d'une stratégie complète.

* * *

Ayant donné un aperçu de l'utilisation des règles du français dans notre système et du formalisme proposé pour transcrire les stratégies imaginées, nous passons dans le chapitre suivant à leur mise en pratique.

5

Mise en œuvre des stratégies et expérimentation

Ce chapitre est consacré à la mise en pratique des idées et représentations exposées précédemment. Nous nous attacherons à montrer comment écrire des règles de contrôle traduisant les stratégies que nous proposons de mettre en œuvre. Pour simplifier notre exposé, nous présentons isolément, de façon un peu arbitraire, un certain nombre de phénomènes stratégiques que nous voulons mettre en avant. Nous commençons par exposer les stratégies globales pour l'analyse d'un énoncé et la détection des fautes qu'il contient. Nous détaillons ensuite les stratégies sur les opérations de base comme l'analyse morphologique et l'analyse syntaxique à partir de la représentation des connaissances que nous avons proposée. Nous poursuivons par la présentation de stratégies de reprise. Nous terminons enfin en donnant des informations sur les résultats obtenus en faisant tourner le système avec des stratégies différentes sur un échantillon d'énoncés.

5.1 Organisation générale de la vérification orthographique

Cette section est consacrée à la description de stratégies générales pour l'analyse des énoncés. Nous renverrons régulièrement aux autres sections du chapitre pour les détails concernant les stratégies plus locales.

L'explication des fautes contenues dans une phrase repose sur un examen du fonctionnement des règles d'analyse. Les connaissances sur la langue ont été données de façon distincte pour ce qui concerne la morphologie et la syntaxe. Ceci nous permet de contrôler de façon séparée la bonne orthographe d'un mot considéré isolément, et du même mot en accord avec son contexte. D'autre part, la simple constatation que les règles se bloquent suffit à déceler la présence d'anomalies. De ce fait, la détection de la simple présence de

fautes, sans vouloir donner plus ample précision, peut s'écrire par la règle de stratégie suivante :

(800	*stratégie*	
(Si	*(objectif (bien_ecrit ?n enonce)))*	*si l'objectif est de vérifier que l'énoncé est correct*
(Alors	*(res_avant (bien_ecrit mots_isolés enonce))*	*alors chercher à vérifier que les mots pris isolément sont bien écrits*
	(res_avant (bien_ecrit ?n mots_en_contexte))))	*et chercher à vérifier que ces mots pris dans leur contexte sont également bien écrits*[1]

Chacun des deux problèmes posés peut encore se décomposer en deux traitements consécutifs :

(801	*stratégie*	
(Si	*(objectif (bien_ecrit mots_isolés enonce)))*	*si l'objectif est de vérifier que les mots pris isolément sont bien écrits*
(Et_Si	*(distribue ?x mots_de_enonce)*	*et si ?x est un mot de l'énoncé*
	(chaîne ?y ?x))	*et que ?y est le "mot de la langue" correspondant à ?x*
(Alors	*(r_a*[2] *(nature ?n ?y))*	*alors chercher à reconnaître ?y*
	(r_a (impression_bilan ?n ?y))))	*puis effectuer le bilan de cette reconnaissance*

(802	*stratégie*	
(Si	*(objectif (bien_ecrit ?n mots_en_contexte)))*	*si l'objectif est de vérifier que les mots examinés dans leur contexte sont bien écrits*
(Alors	*(r_a (anal_synt ?n enonce))*	*alors effectuer une analyse syntaxique de l'énoncé*
	(r_a (impression_bilan ?n enonce))))	*puis effectuer le bilan des informations obtenues*

1. Le paramètre *?n* qui apparaît dans l'objectif intermédiaire (`bien_ecrit ?n mots_en_contexte`) est inroduit dans le §5.1.1.

2. Pour alléger le texte des règles, la fonction `res_avant` est dorénavant notée `r_a`

Nous avons pris la convention de souligner les fonctions de base comme distribue ou chaîne utilisées dans le texte des règles.

La fonction (distribue ?x l) instancie successivement ?x par chacun des éléments de l.

La fonction (chaîne ?y ?x) extrait du mot de la phrase ?x le mot de la langue qui lui correspond ?y (par exemple on extrait *"bateau"* de *("bateau" 3)*).

Tout comme l'objet **enonce** est un élément de la base de travail et représente l'énoncé à analyser, la liste **mots_de_enonce** est une variable globale qui contient la liste des mots de l'énoncé.

Vérifier qu'un mot ?x de la phrase est bien écrit, sans tenir compte de son contexte d'utilisation, revient à reconnaître si ce mot existe dans la langue. On peut ramener ce problème à celui de la recherche d'hypothèses grammaticales sur le mot de la langue ?y associé à ?x, c'est à dire à celui de l'instanciation du prédicat (nature ?n ?y) qui est le premier objectif intermédiaire de la règle 801.

5.1.1 Stratégie d'analyse multiple

Une façon classique d'effectuer les analyses consiste à rechercher toutes les solutions possibles : *stratégie d'analyse multiple.* Pour cette stratégie :

- la reconnaissance d'un mot consiste à examiner s'il existe tel quel dans le lexique, et à savoir si c'est aussi une variante morphologique d'une autre mot du lexique, ce que l'on fait en écrivant la règle de stratégie :

(803	*stratégie*	
(Si	*(objectif (nature ?n ?x)))*	*si l'objectif est la reconnaissance de ?x*
(Alors	*(recherche lexique ?x)*	*alors rechercher ?x dans le lexique*
	(r_a (anal_morpho ?n ?x))))	*et chercher à effectuer une analyse morphologique de ?x*

La mise en œuvre de l'opération d'analyse morphologique et des stratégies associées est détaillée dans la section 5.2.1.

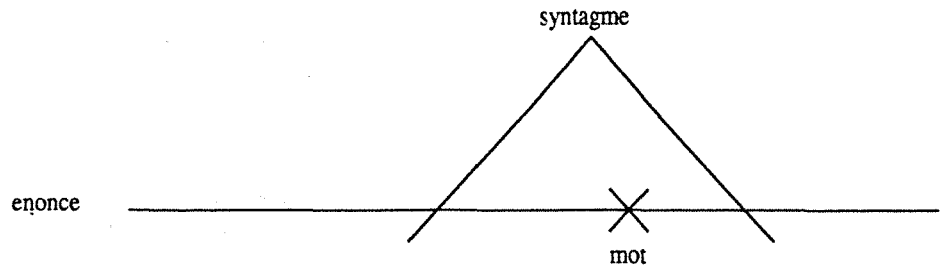
exemple 1

"les descriptions des modules son_(t) dépendantes de la syntaxe des langage_(s)"

Base d'hypothèses après application de la règle de stratégie 803 :

("de"	*(1 (nature prep)))*
("des"	*(1 (nature det) (nombre plur))*
	(2 (nature prep) (nombre plur)))
("descriptions"	*(1 (nature nom) (genre fem) (nombre plur)))*
("la"	*(1 (nature det) (genre fem) (nombre sing))*
	(2 (nature part-prev-cod) (genre fem) (nombre sing)))
("langage"	*(1 (nature nom) (genre masc) (nombre sing)))*
("les"	*(1 (nature det) (nombre plur))*
	(2 (nature part-prev-cod) (nombre plur)))
("modules"	*(1 (nature nom) (genre masc) (nombre plur))*
	(2 (nature verbe) (nombre sing) (personne 2)))
("son"	*(1 (nature det) (genre masc) (nombre sing))*
	(2 (nature nom) (genre masc) (nombre sing)))
("syntaxe"	*(1 (nature nom) (genre fem) (nombre sing)))*

- l'examen des mots en contexte consiste à étudier :
  - d'une part si le mot est bien accordé avec ses voisins. Pour cela il suffit que l'on trouve un arbre syntaxique recouvrant une entité significative de la phrase pour assurer un diagnostic fiable.



C'est une raison pour préférer une analyse syntaxique ascendante. Il y en a d'autres (cf §5.1.5).

  - d'autre part si l'énoncé est bien bâti. On sort du strict cadre de la vérification orthographique pour contrôler également la construction des phrases. Dans

ce cas, tout l'énoncé doit pouvoir être généré à partir d'un seul axiome de la grammaire. C'est l'objectif qu'il faut atteindre pour minimiser les oublis. Il permet entre autres d'aider à répondre au problème précédent en localisant les constituants syntaxiques de l'énoncé dans lesquels vérifier les accords. Mais il peut s'avérer trop complexe ou trop coûteux et amener à se contenter d'analyses partielles.

Pour les phrases simples données en exemple, il n'y a pas de raison de ne pas essayer de prime abord une analyse syntaxique complète de la phrase et de demander plus précisément dans la règle 800 la recherche de l'objectif intermédiaire `(bien_ecrit prop mots_en_contexte)` où `prop` signifie que l'on cherche à vérifier que l'énoncé a une structure de *proposition.*

Les traitements à déclencher pour effectuer l'analyse syntaxique de l'énoncé dépendent de la stratégie que l'on souhaite suivre. Nous avons basé notre exposé sur les stratégies applicables sur la représentation des connaissances. L'ensemble des constituants de l'énoncé est alors obtenu par un cycle d'inférences sur les règles syntaxiques.

(804	*stratégie*	
(Si	*(objectif (anal_synt ?s enonce)))*	*si l'objectif est l'analyse syntaxique de l'énoncé pour arriver à reconnaître qu'il a pour structure ?s*
(Alors	*(r_a (définir mots_de_enonce))*	*alors définir tout d'abord chacun des mots de l'énoncé*
	(cycle_avant l_regles_synt enonce ?s)))	*puis effectuer un cycle d'inférences sur les règles syntaxiques pour obtenir la structure* (nature ?s enonce)

Il apparaît un problème occulté jusqu'à présent qui concerne le passage des hypothèses du mot de la langue, sur lequel opèrent le lexique et les règles morphologiques, au mot de la phrase sur lequel travaillent les règles syntaxiques. La propagation de ces hypothèses doit être déclenchée de façon explicite comme c'est fait dans la règle 805.

(805	*stratégie*	
(Si	*(objectif (définir ?l)))*	*si l'objectif est de définir chaque mot de la liste ?l*
(Et_Si	*(distribue ?x ?l)*	*et si ?x est un élément de ?l*
	(chaîne ?y ?x))	*et que ?y est le mot de la langue correspondant à ?x*
(Alors	*(attribuer (nature ?n ?y) ?x)))*	*alors propager les hypothèses sur la nature de ?y à ?x*

On obtient sur l'exemple 1 :

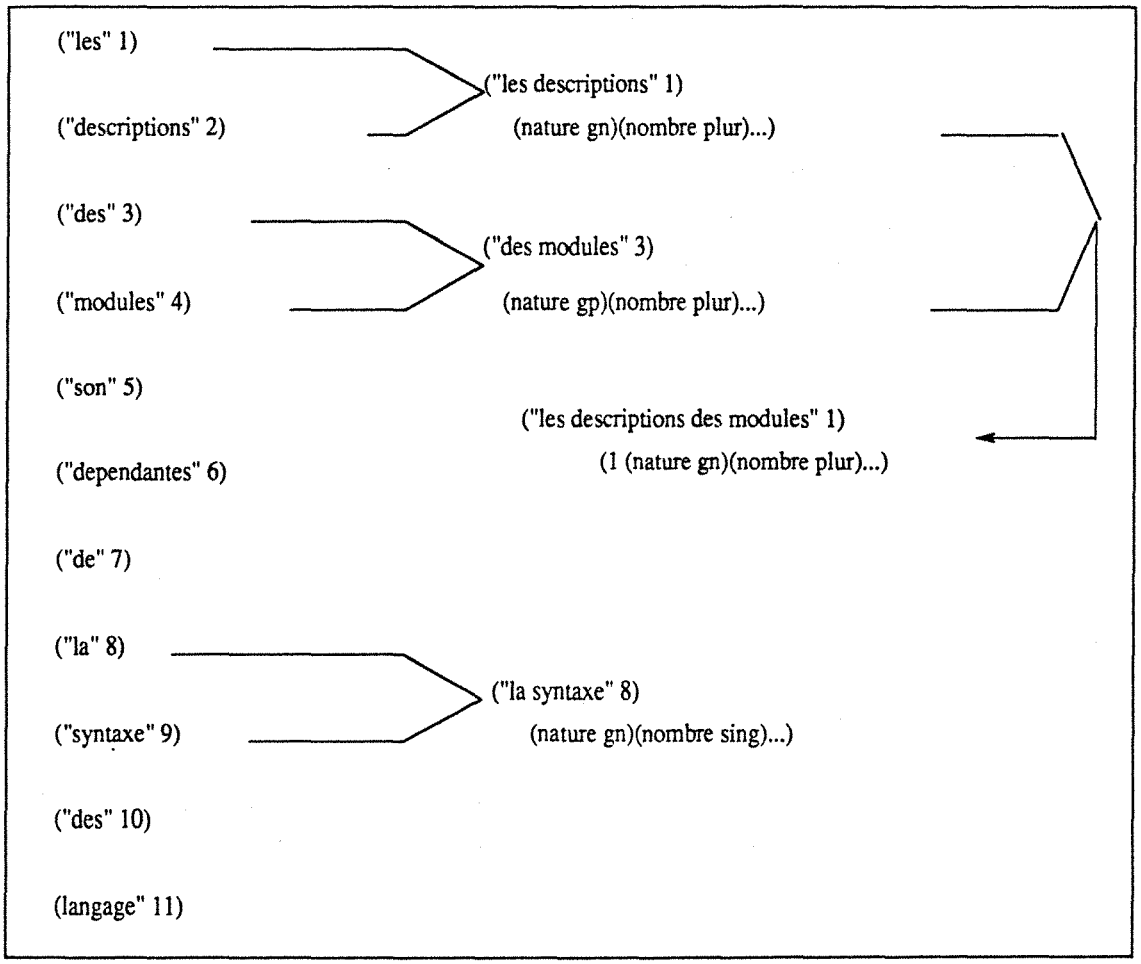


Figure 5.1. Premiers résultats d'une analyse syntaxique tolérante sur *"les descriptions des modules son dépendantes de la syntaxe des langage"*.

On est en situation d'échec, ce qui n'a rien pour nous étonner. Seulement, le bilan est assez négatif :

- il a fallu essayer tous les cas possibles, ce qui risque d'être déraisonnable pour des phrases non triviales ;
- on n'a surtout pas d'espoir de récupérer des informations pertinentes pour expliquer les fautes si on devait le faire. Aucune information sur les causes des blocages rencontrés n'a en effet été récupérée.

C'est pour remédier à ce genre d'inconvénient que l'on a recours à d'autres stratégies générales complémentaires :

- l'analyse immédiate (§5.1.2) ;
- l'analyse tolérante (§5.1.3).

Ces stratégies n'utilisent pas de connaissances particulières du français. Elles mettent en œuvre des mécanismes d'analyse généraux et seront complétées par d'autres stratégies plus locales davantage liées aux connaissances sur la langue. Nous montrons ci-après comment de telles stratégies peuvent être mises en place avec la représentation proposée pour exprimer les stratégies.

5.1.2 Stratégie d'analyse immédiate

Le principe de cette stratégie consiste à :

- exécuter en premier les traitements les plus simples ;
- s'arrêter provisoirement (cf §5.1.4) sur la première solution satisfaisante.

Ainsi la règle de vérification de l'orthographe d'un mot considéré isolément est scindée en deux règles distinctes et ordonnées par complexité croissante.

(806	*stratégie*	
(Si	*(objectif (nature ?n ?x)))*	*si l'objectif est la reconnaissance de ?x*
(Alors	*(recherche lexique ?x)))*	*alors rechercher ?x dans le lexique*

Ce qui donne sur l'exemple 1 :

```
( "de"        (1 (nature prep)))
( "des"       (1 (nature det) (nombre plur))
              (2 (nature prep)(nombre plur)))
( "la"        (1 (nature det) (genre fem) (nombre sing))
              (2 (nature part-prev-cod) (genre fem) (nombre sing)))
( "langage"   (1 (nature nom) (genre masc) (nombre sing)))
( "les"       (1 (nature det) (nombre plur))
              (2 (nature part-prev-cod) (nombre plur)))
( "son"       (1 (nature det) (genre masc) (nombre sing))
              (2 (nature nom) (genre masc) (nombre sing)))
( "syntaxe"   (1 (nature nom) (genre fem) (nombre sing)))
```

On trouve immédiatement dans le lexique :

- les mots outils dont toutes les flexions ont été recensées dans le lexique. Ces mots sont d'un emploi suffisamment fréquent pour éviter d'avoir recours au traitement morphologique, ils ont d'aileurs fait l'affaire de plusieurs recensements dont [Catach 84b]. De plus, il est intéressant de les repérer rapidement pour éventuellement poser des jalons pour la suite des traitements.
- les mots utilisés sous leur forme canonique dans la phrase.

```
(807    stratégie
(Si     (objectif (nature ?n ?x)))          si l'objectif est la reconnaissance de ?x
(Alors  (r_a (anal_morpho_imm ?n ?x))))     alors chercher à effectuer une analyse morphologique de ?x avec stratégie immédiate
```

La base d'hypothèses se trouve ainsi complétée :

```
( "dépendantes" (1 (nature adj) (genre fem) (nombre plur)))
( "descriptions" (1 (nature nom) (genre fem) (nombre plur)))
( "modules"     (1 (nature nom) (genre masc) (nombre plur)))
```

On ne se rend pas compte ici de l'économie de traitement réalisé, hormis l'absence de la deuxième solution pour *"modules"* (verbe). Le fait de s'arrêter à la première solution réduit cependant de façon non négligeable l'ampleur des traitements morphologiques (cf §5.2.1).

5.1.3 Stratégie d'analyse tolérante

Cette stratégie autorise un emploi permissif des règles. Elle consiste à pratiquer consciemment une impasse sur les conditions les moins strictes.

L'ordre de vérification des conditions est le suivant :

- *SI*
- *ET_SI*
- *SAUF*
- *SOUS_RESERVE*

Dans une première expérimentation, nous n'avons envisagé de pratiquer l'impasse que sur les clauses *Sous_Réserve* (cf §5.3.2) et *Sauf* (cf §5.2.2). La vérification de leurs conditions est respectivement contrôlée par les paramètres `confirm` et `prof` des fonctions d'inférences comme nous l'avons modifié dans la règle 804. La syntaxe complète de la fonction `cycle_avant` est donnée §5.3.3.

(804	*stratégie*	
(Si	*(objectif (anal_synt ?s enonce)))*	*si l'objectif est l'analyse syntaxique de l'énoncé pour arriver à reconnaître qu'il a la structure ?s*
(Alors	*(r_a (définir mots_de_enonce))*	*alors définir tout d'abord chacun des mots de l'énoncé*
	(cycle_avant l_regles_synt enonce ?s confirm prof ...)))	*puis effectuer un cycle d'inférences sur les règles syntaxiques pour obtenir la structure* (nature ?s enonce)

`confirm`	booléen	à vrai si les règles doivent être confirmées pour pouvoir être applicables ; à faux si on autorise l'application des règles même si elle ne sont pas confirmées.
`prof`	condition	avant la validation de toute règle, `prof` est évaluée ; si elle est vérifiée, on bloque la validation de la règle qui est alors ignorée. Cette condition peut porter par exemple sur la profondeur relative de la règle dans la hiérachie.

Ainsi, les accords mentionnés généralement dans les clauses *Sous_Réserve* (cf §3.4.2) ne sont pas forcément vérifiés pour élaborer une ébauche d'arbre syntaxique. Du coup, en mettant `confirm` à faux pour l'exemple déroulé, on parvient à obtenir une analyse qui recouvre davantage l'énoncé. Les inférences effectuées s'avèrent en effet vérifiées au niveau global, alors qu'elles ne le sont pas au niveau local.

Cependant, il est dans un grand nombre de cas trop permissif de développer systématiquement une analyse tolérante. Il est souvent préférable de n'autoriser ces approximations qu'en cas de nécessité (cf §5.3.2).

5.1.4 Reprise

Lors du constat d'échec sur l'analyse de l'exemple 1, plusieurs types de reprise peuvent être envisagés.

Reprise des traitements arrêtés temporairement

Tout d'abord, on ne peut pas conclure à une anomalie sur la structure de l'énoncé tant qu'on n'a pas complété l'ensemble de ces premières solutions apportées par une stratégie immédiate (backtrack).

De façon générale, lorsqu'un échec est constaté sur la recherche d'un objectif, ici `(anal_synt prop enonce)`, on cherche à obtenir de nouvelles hypothèses qui permettraient cette fois à l'analyse d'aboutir. On redéclenche alors la règle sur laquelle l'échec a eu lieu (ici la règle 804 page 97) de façon à relancer les sous-objectifs intermédiaires `(definir mots_de_enonce)` pour en obtenir de nouvelles informations.

Une nouvelle application de la règle 805 pour la recherche de l'objectif `(definir mots_de_enonce)` n'apporte cependant rien si aucune nouvelle hypothèse grammaticale sur les mots de la langue n'est venue complétée la base d'hypothèses. Pour ce faire, on

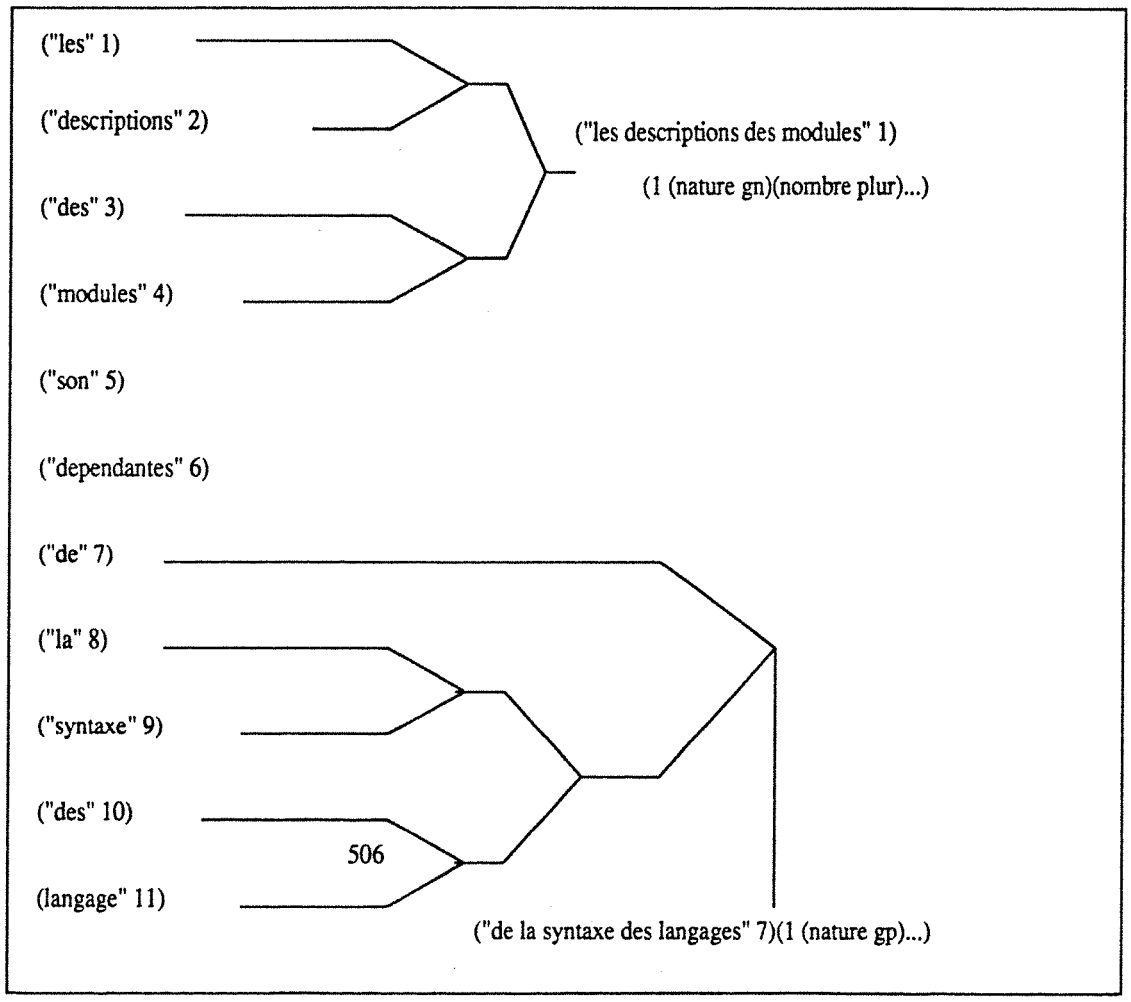


Figure 5.2. Premiers résultats d'une analyse syntaxique tolérante sur *"les descriptions des modules son dépendantes de la syntaxe des langage"*.

doit définir une action de reprise pour redéclencher la recherche de nouvelles hypothèses grammaticales sur les mots de la phrase. La règle de reprise 840 associée à la règle de stratégie 805 le fait successivement pour chacun des mots de la phrase, à charge à de stratégies moins "triviales" de donner les moyens de repérer les hypothèses les plus suspectes à remettre en cause en priorité (cf §5.4).

(840	*reprise*	
(Si	*(objectif (definir ?l)))*	*si l'objectif est de redéfinir les éléments de ?l*
(Et_Si	*(distribue ?x ?l)*	*et si ?x est un mot de la liste ?l*
	(chaîne ?y ?x))	*et que ?y est le mot de la langue correspondant au mot ?x*
(Alors	*(reprendre_objectif (nature ?n ?y))*	*alors reprendre la recherche de l'objectif* (nature ?n ?y)
	(attribuer (nature ?n ?y) ?x)))	*et attribuer à ?x les hypothèses obtenues sur la nature de ?y*

La fonction `reprendre_objectif` est du même type que la fonction `res_avant`. C'est une fonction du système fournie pour l'écriture des règles de stratégie qui permet de redéclencher de façon explicite la recherche d'un objectif temporairement arrêté au cours des étapes précédentes.

Cette reprise résumée par la figure 5.3[3] n'offre cependant pas la solution attendue sur la structure de la phrase.

Cette première catégorie de reprise peut être généralisée à tous les objectifs, la seconde présentée ci-dessous est plus spécifique. Elle correspond à une action de correction orthographique proprement dite et s'appuie sur l'hypothèse que l'échec est dû à un mot mal écrit dans la phrase et pour lequel on veut proposer des corrections possibles.

Correction

En cas de blocage, si on fait l'hypothèse qu'il y a une faute sur un mot de la phrase, il n'est pas question de proposer des corrections pour chaque mot de la phrase. Il faut pouvoir repérer les points d'anomalie sur lesquels focaliser les traitements.

3. Les objectifs soumis au système sont encadrés d'une ellipse, les hypothèses nouvellement inférées d'un rectangle.

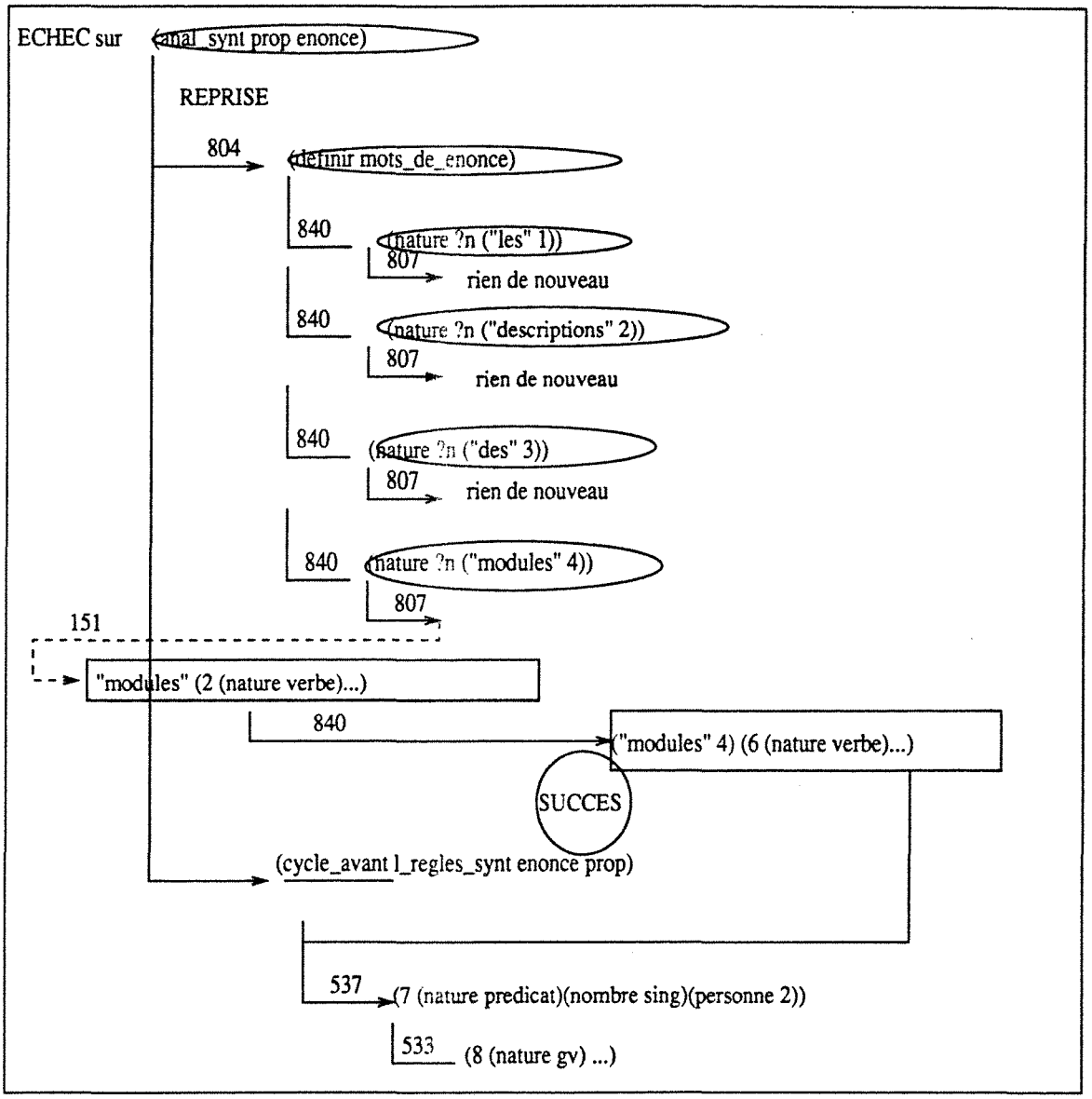


Figure 5.3. Reprise après échec d'une analyse immédiate sur la phrase *"les descriptions des modules son dépendantes de la syntaxe des langage"*.

"Le mot ("son" 5*) est suspect"*
"On peut le remplacer par "sont" *ou par ..."*

Les règles de reprise de ce type auront alors la forme suivante, sachant que `mot_suspect` représente une fonction quelconque pour repérer les mots suspects et `correction` une fonction quelconque de correction orthographique :

(...	*reprise*	
(Si	*(objectif (definir ?l)))*	*si l'objectif est de redéfinir les éléments de ?l*
(Et_Si	*(*mot_suspect *?x ?l)*	*et si ?x est un mot sur lequel on suspecte une anomalie*
	(chaîne ?y ?x))	*et que ?y est le mot de la langue correspondant au mot ?x*
(Alors	(correction *?z ?y)*	*alors proposer une correction ?z de ?y*
	(r_a (nature ?n ?z))	*rechercher des hypothèses sur ?z*
	(attribuer (nature ?n ?y) ?x)))	*et les attribuer à ?x*

5.1.5 Explicitation des erreurs

En cas d'erreurs, les explications à apporter s'appuient sur les informations recueillies au cours de l'analyse :

- blocages dûs à constituants non reconnus ;
- non respect des contraintes de règles appliquées.

(809	*stratégie*	
(Si	*(objectif (impression_bilan ?n ?x)))*	*si l'objectif est d'imprimer le bilan de l'analyse sur le mot ?x*
(Et_Si	*(non (nature ?n ?x)))*	*et si aucune hypothèse grammaticale n'a été obtenue sur ?x*
(Alors	*(écrire "le mot" ?x "n'a pas été reconnu")))*	*alors signaler l'anomalie*

On regarde par l'intermédiaire de la règle 809 s'il existe une hypothèse grammaticale sur le mot et on signale une anomalie dans le cas contraire.

(811	*stratégie*	
(Si	*(objectif (impression_bilan ?s enonce)))*	*si l'objectif est d'imprimer le bilan de l'analyse de l'énoncé*
(Et_Si	*(non (nature ?s enonce)))*	*et si on n'a pas pu déterminer de structure syntaxique pour l'énoncé*
(Alors	*(ecrire "la structure syntaxique de l'énoncé)*	*alors signaler l'anomalie*
	(écrire "n'a pas pu être déterminée")))	

Cependant, le diagnostic de structure inconnue de la phrase n'est pas suffisant. En effet, lorsque l'analyse effectuée est tolérante, il convient de signaler les solutions pour lesquelles il y a hypothèse d'erreur. Cela revient à signaler les règles appliquées pour aboutir à cette solution et dont toutes les conditions ne sont pas respectées. Si la tolérance autorisée ne porte que sur les clauses *Sauf* et *Sous_Réserve*, il faut signaler les règles dont les exceptions n'ont pas été respectées (cf §5.2.2) et celles qui ont été validées mais non confirmées.

(812	*stratégie*	
(Si	*(objectif (impression_bilan ?n enonce)))*	*si l'objectif est d'imprimer le bilan de l'analyse de l'énoncé*
(Et_Si	*(distribue ?r l_regles_non_confirmees)*	*et si ?r est une règle de l_regles_non_confirmees*
	(condition_non_verifiee ?c ?r))	*et que ?c est la condition non vérifiée de ?r*
(Alors	*(ecrire "la règle " ?r " a été mal appliquée")*	*alors signaler l'anomalie sur cette règle*
	(ecrire "la condition " ?c " n'est pas vérifiée")))	

`l_regles_non_confirmees` : liste construite en cas de stratégie tolérante au fur et à mesure des inférences sur les règles, et qui recense les règles validées mais non confirmées qui ont été appliquées,

Fonction

`(condition_non_verifiee ?c ?r)` extrait d'une règle `?r` de la liste `l_regles_non_confirmees` la condition non vérifiée `?c`.

De l'exemple déroulé ci-dessus, on obtient la liste `l_regles_non_confirmees` :

((506 ((("des" 10) . 3)(("langage" 11) . 5)) (nombre sing ("des" 10))))

La règle 506 est décrite page 202. Le résultat de la fonction `condition_non_verifiee` est : *(nombre sing (* "des" 10*))*.

D'où le diagnostic :

La règle 506 a été mal appliquée :
La condition (nombre sing (*"des" 10*)) *n'est pas vérifiée*

Figure 5.4.

On remarque ici l'avantage d'une analyse ascendante qui permet de suggérer des solutions, même si la structure complète de la phrase n'a pu être déterminée.

5.1.6 Récapitulatif

Une stratégie immédiate et tolérante, peut être définie, si on fait abstraction de l'opération d'analyse morphologique et des paramètres qu'elle fait intervenir, par les règles et les paramètres suivants :

`base_de_strategie` =
((800) (801) (802) (804) (805 840) (806) (807) (809) (811) (812))

`confirm` = *faux*

L'application d'une telle stratégie sur l'exemple 1 donne à la base de travail les états successifs schématisés dans les figures 5.2 puis 5.3 et au diagnostic de la figure 5.4.

* * *

Nous allons maintenant détailler la mise en œvre de stratégies sur les opérations d'analyse morphologique et d'analyse syntaxique. Un de nos objectifs est de montrer comment exploiter la représentation progressive des connaissances linguistiques, mais aussi d'illustrer la mise en place d'autres stratégies classiques par les règles de contrôle.

5.2 Analyse morphologique

Nous entendons par *analyse morphologique* l'opération de recherche d'hypothèses grammaticales caractérisant une forme fléchie donnée. Nous présentons ici les possibilités offertes par notre représentation des connaissances pour définir des stratégies sur l'analyse morphologique. Nous avons vu dans le chapitre 4 que les règles morphologiques utilisées sous leur forme initiale ne permettent pas de répondre directement et de façon raisonnable à ce problème. Pour éviter l'inconvénient du chaînage arrière qui serait nécessaire et de la combinatoire qui en découle, nous avons proposé un traitement de nature compilatoire (cf §4.1.2) associant à chaque désinence les règles morphologiques utiles pour retrouver l'information.

5.2.1 L'opération d'analyse morphologique

Extraire l'information pertinente

Dans la première étape de cette opération il faut extraire du mot l'information pertinente. La solution adoptée consiste à procéder flexion par flexion. Sur le mot *"nouvelles"* par exemple, on n'extrait dans un premier temps que la désinence *"s"* renvoyant aux règles sur la construction du pluriel des noms et c'est lors de l'analyse du radical *"nouvelle"* que l'on considère la désinence *"elle"* renvoyant aux règles sur la construction du féminin.

La fonction auto_morpho rend pour un mot donné, au vu de sa structure, la liste des radicaux et des règles morphologiques à partir desquels il a pu être construit.

Sur le mot *"descriptions"* de l'exemple 1, elle fournit les solutions suivantes, présentées par couple *(numéro de règle, radical associé)* :

```
((50  (mot . "description"))   % règle sur le pluriel des noms %
 (70  (mot . "description"))   % règle sur le pluriel des adjectifs %
 (80  (mot . "description"))   % règle sur le pluriel des participes passés %
 (165 (mot . "descriptier"))   % règle sur la conjugaison au présent des verbes en "er" %
 (185 (mot . "descripter"))    % règle sur la conjugaison à l'imparfait des verbes en "er" %
```

Les hypothèses sur le mot seront ensuite obtenues par inférence sur les règles initiales sélectionnées et applicables.

Inférences

La fonction `cycle_arriere` procède aux inférences en chaînage arrière.

(813	*stratégie*	
(Si	*(objectif (anal_morpho ?n ?x)))*	*si l'objectif est d'effectuer l'analyse morphologique de ?x*
(Et_Si	*(egal ?l (auto_morpho ?x ?n)))*	*et si la liste des règles morphologiques ayant pu permettre de construire ?x est stockée dans ?l*
(Alors	*(cycle_arriere ?l)*	*alors déclencher chacune de ces règles en chaînage arrière*

La fonction `(egal var val)` instancie la variable `var` le résultat de l'évaluation de `val`.

On obtient sur *"descriptions"* la trace schématisée en figure 5.5.

Lorsque le processus d'inférence en chaînage arrière soumet au système la vérification d'une condition de règle *(nature ?n ?x)*, la catégorie grammaticale symbolisée par *?n* est connue. Par exemple, pour valider la règle 50 sur *"description"*, il faut vérifier la condition *(nature nom "description")*. Ceci explique a posteriori la présence du paramètre `?n` dans les objectifs de la forme `(nature ?n ?x)` ainsi que dans la fonction `auto_morpho`. Ce paramètre permet de contraindre l'analyse à la seule catégorie recherchée.

exemple :

Le résultat de *(auto_morpho "descriptions" verbe)* est réduit à :

((165 (mot . "descriptier*)) % règle de conjugaison au présent des verbes en* "er" %
(185 (mot . "descripter*)) % règle de conjugaison à l'imparfait des verbes en* "er" %

5.2.2 Stratégies d'analyse morphologique

Certains éléments de stratégie peuvent être apportés au niveau de l'analyse morphologique. Nous en décrivons quelques-uns ci-dessous qui jouent aussi bien sur la premi're étape de sélection que sur la second étape d'inférences de l'analyse morphologique.

a) Analyse morphologique immédiate

Pour rentrer dans le cadre d'une stratégie immédiate (cf §5.1.2) une variante notée `cycle_arriere_popa` de la fonction `cycle_arriere` est paramétrée par l'objectif à at-

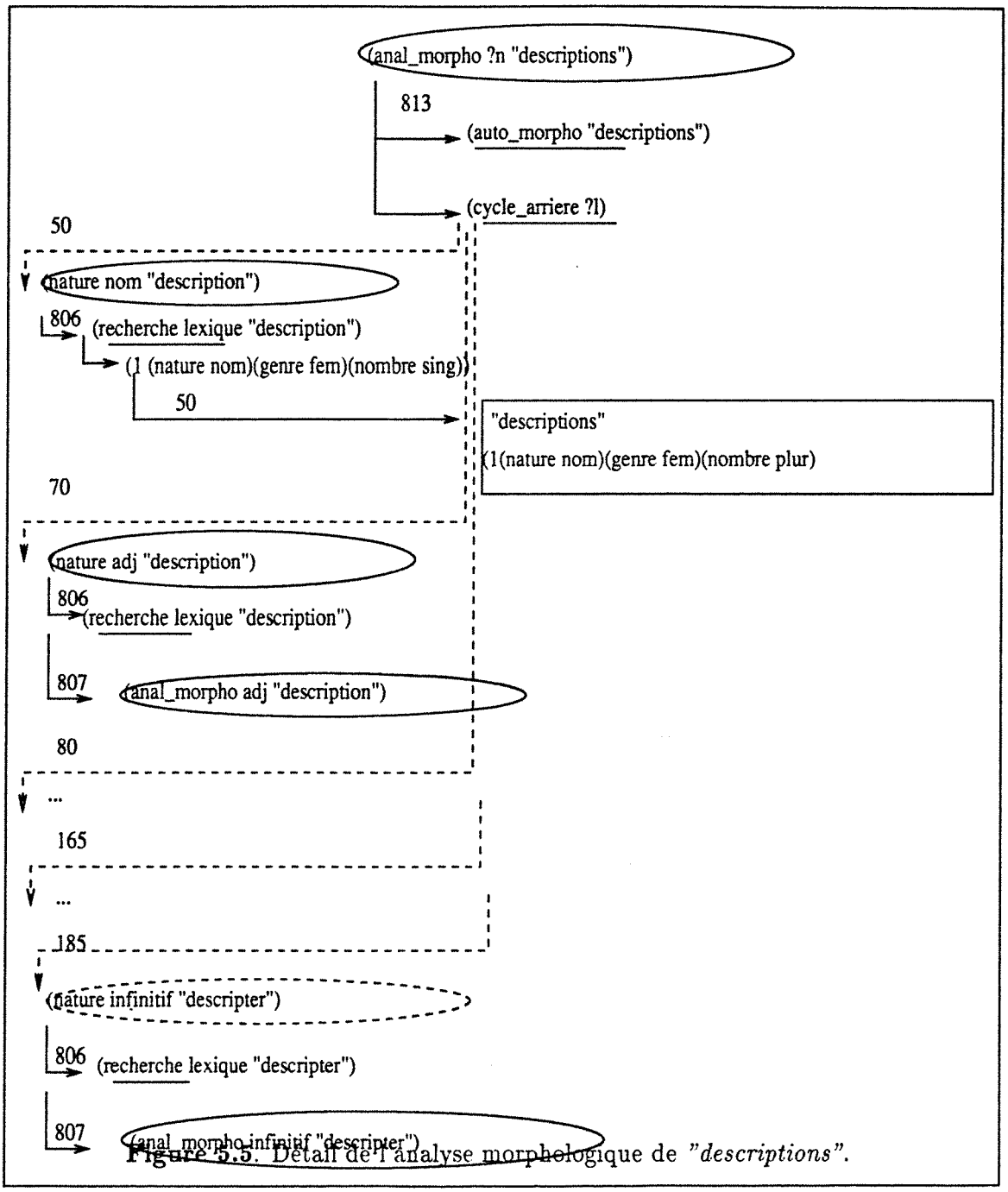


Figure 5.5. Détail de l'analyse morphologique de *"descriptions"*.

teindre. Les inférences effectuées par cycle_arriere_popa sont arrêtées temporairement dès que ce dernier est réalisé.

(814	*stratégie*	
(Si	*(objectif (anal_morpho_imm ?n ?x)))*	*si l'objectif est d'effectuer l'analyse morphologique de ?x*
(Et_Si	*(egal ?l (auto_morpho ?x ?n)))*	*et si la liste des règles morphologiques ayant pu permettre de construire ?x est stockée dans ?l*
(Alors	*(cycle_arriere_popa ?l (nature ?n ?x))*	*alors déclencher chacune de ces règles en chaînage arrière jusqu'à réalisation de l'objectif* (nature ?n ?x*)*

L'analyse morphologique de *"descriptions"* est réduite d'autant :

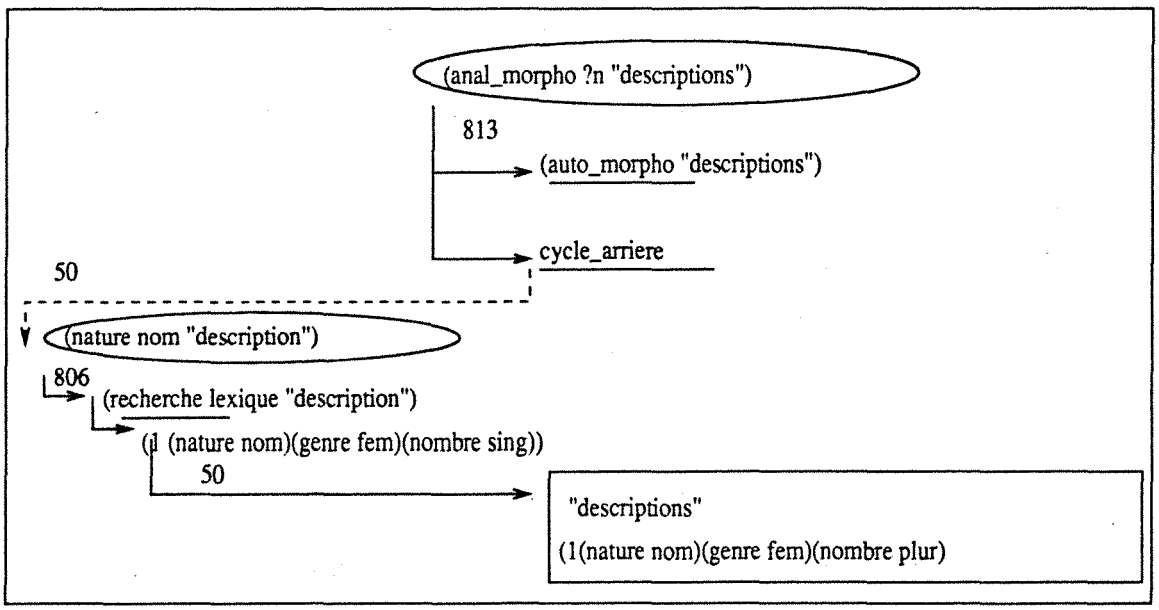


Figure 5.6. Détail d'une analyse morphologique immédiate de *"descriptions"*.

On peut ici se rendre compte de l'économie de traitement réalisé et remarquer :

- que l'ordre dans lequel les règles sont sélectionnées influe sur les traitements développés. L'économie n'aurait pas été la même si on avait ordonné les hypothèses sur la décomposition morphologique de *"descriptions"* de façon inverse (verbe, adjectif, nom) ;

- la stratégie en profondeur d'abord pour le chaînage arrière appliquée n'est pas la seule stratégie possible.

b) Ordonnancement des hypothèses

Un ordre classiquement suivi pour l'analyse morphologique est celui qui joue sur la longueur de la désinence de la forme fléchie. On peut en effet supposer qu'une désinence plus longue correspond a priori à une hypothèse plus forte (même si on remarque que pour *"descriptions"* cela ne s'avère pas vrai). Cette stratégie peut être discutée par le fait que les désinences plus longues correspondent en règle générale aux verbes, et que sur l'ensemble des mots analysés, on rencontre moins de verbes que de noms ou d'adjectifs. On peut préférer pour cela une stratégie qui ordonne les règles par longueur de désinence décroissante.

Il est possible également d'ordonner les hypothèses morphologiques en fonction des catégories grammaticales que l'on souhaite favoriser : dans l'exemple déroulé sur *"descriptions"* en figure 5.6, à longueur de désinence égale, l'hypothèse sur le nom a été (arbitrairement) privilégiée par rapport aux hypothèses sur l'adjectif et le participe passé.

Mise en œuvre

L'automate morphologique (cf §4.1.2) est mis en place par l'intermédiaire de *règles inverses*. Nous expliquons leur rôle après avoir décrit la forme finale de la fonction `auto_morpho` qu'elles permettent de paramétrer.

fonction `auto_morpho`
syntaxe : `(auto_morpho l_regles_inverses mot cat ordre)`

`l_regles_inverses`	*liste de règles à partir desquelles effectuer la décomposition morphologique*
`mot`	*mot à analyser*
`cat`	*catégorie grammaticale recherchée pour mot ou nil*
`ordre`	*= "croiss" si on veut ordonner les règles par longueur de désinence croissante* *= "decroiss" sinon*

La décomposition *radical-désinence* d'un forme fléchie et le renvoi aux règles morphologiques sont décrits dans des règles appelées *règles inverses* qui sont l'équivalent d'un automate morphologique.

exemples :

(490	*morpho*	
(Si	*(unif mot (conc ?x"s")))*	*si mot s'écrit ?x"s"*
(Alors	*(regles_init (50) ?x)))*	*alors renvoyer à la règle morphologique 50 sur la construction du pluriel des noms sur le radical ?x*

(491	*inverse*	
(Si	*(unif mot (conc ?x"s")))*	*si mot s'écrit ?x"s"*
(Alors	*(regles_init (70) ?x)))*	*alors renvoyer à la règle morphologique 70 sur la construction du pluriel des adjectifs sur le radical ?x*

Ces règles ont été construites de telle façon qu'à une désinence donnée (par exemple *"s"*) et à une catégorie grammaticale donnée (*adj*) correspond une règle inverse (*491*). Par l'ordre des règles inverses passées en paramètre dans la fonction auto_morpho, on définit une priorité sur les catégories grammaticales des hypothèses fournies.

exemple : Si on a une stratégie qui privélégie l'ordre : nom, adjectif, participe passé et verbe et que `ordre` *vaut* `"croiss"`*, on obtient pour* "fermes" *:*

```
(auto_morpho l_regles_inverses "fermes" nil "croiss")
   → (50 (mot . "ferme"))  % pluriel des noms %
   → (70 (mot . "ferme"))  % pluriel des adjectifs %
   → (80 (mot . "ferme"))  % pluriel des participes passes %
   → (151 (mot . "fermer")). % conjugaison des verbes du 1er groupe %
```

c) Stratégie sur le cycle d'inférences

Le cycle d'inférences en chaînage arrière n'est pas limité à une stratégie en profondeur d'abord. Une démarche classique comme la démarche en largeur d'abord peut également être envisagée. Nous décrivons une autre stratégie qui rentre toujours dans le cadre d'une analyse immédiate et qui correspond, pour reprendre les termes de [Farreny 85], à une *stratégie en profondeur d'abord sauf si une règle est immédiatement concluante.*

exemple 2

"une classe hérite des variables de ses classes dominantes"

L'objectif *(nature ?n ?x)* est successivement déclenché pour chacun des mots de la phrase (règle 801). Pour *("classes" 8)*, cela amène à effectuer l'analyse morphologique (règle 807 puis 813)) de *"classes"* schématisée par la figure 5.7.

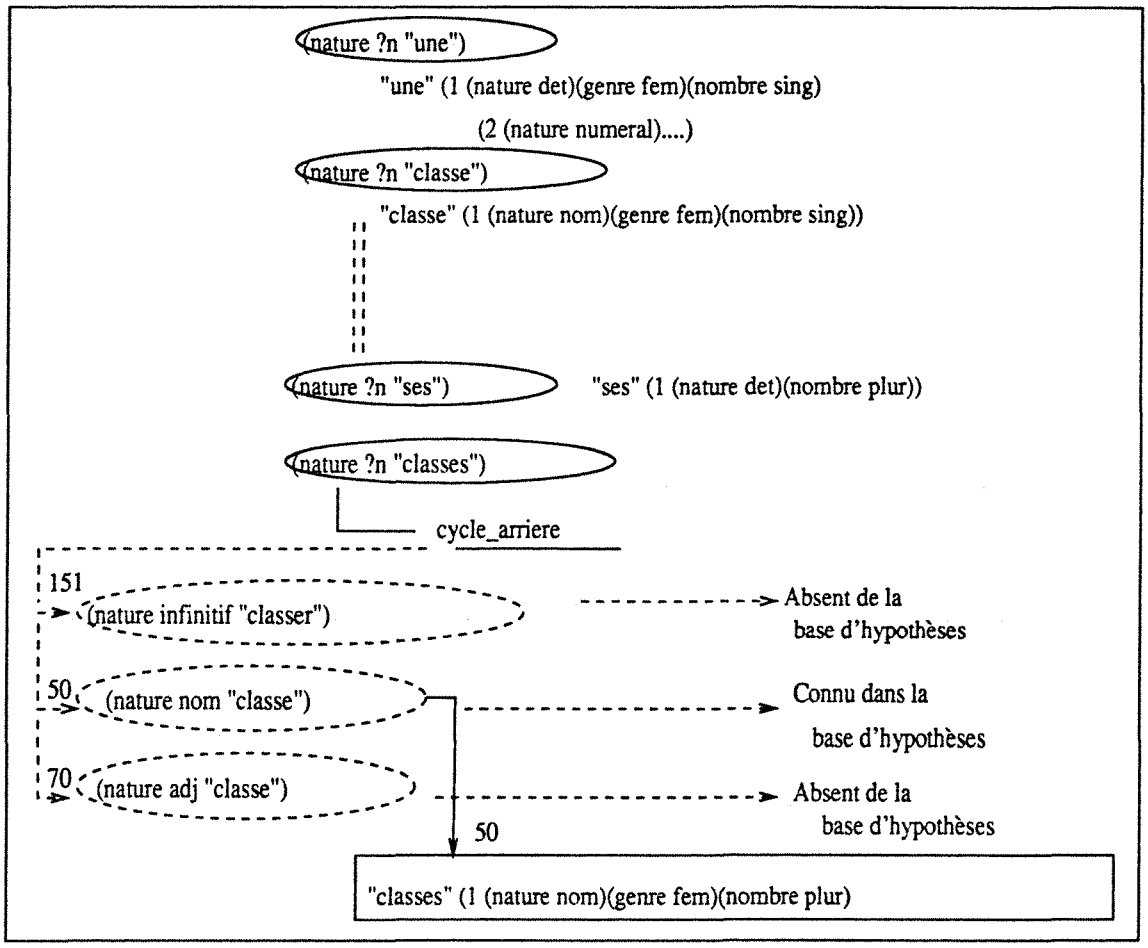


Figure 5.7. Extraits des analyses morphologiques des mots de la phrase *"un classe hérite des variables de ses classes dominantes"*.

La stratégie introduite ci-dessus peut s'expliquer ainsi : si une des règles est immédiatement validée avec les hypothèses actuelles de la base d'hypothèses, elle est appliquée en priorité. La validation des autres règles est mise en attente. Cette stratégie est spécifiée par un paramètre booléen `attente` dans `cycle_arriere_popa`. La syntaxe complète de la fonction `cycle_arriere_popa` est donnée page 117.

d) Stratégie tolérante

Une analyse tolérante au niveau morphologique consiste à déclencher les règles sans présumer des règles d'exceptions qui devraient éventuellement s'appliquer pour les contredire.

exemple 3

"L' environnement de conception génére(ère) des directives"

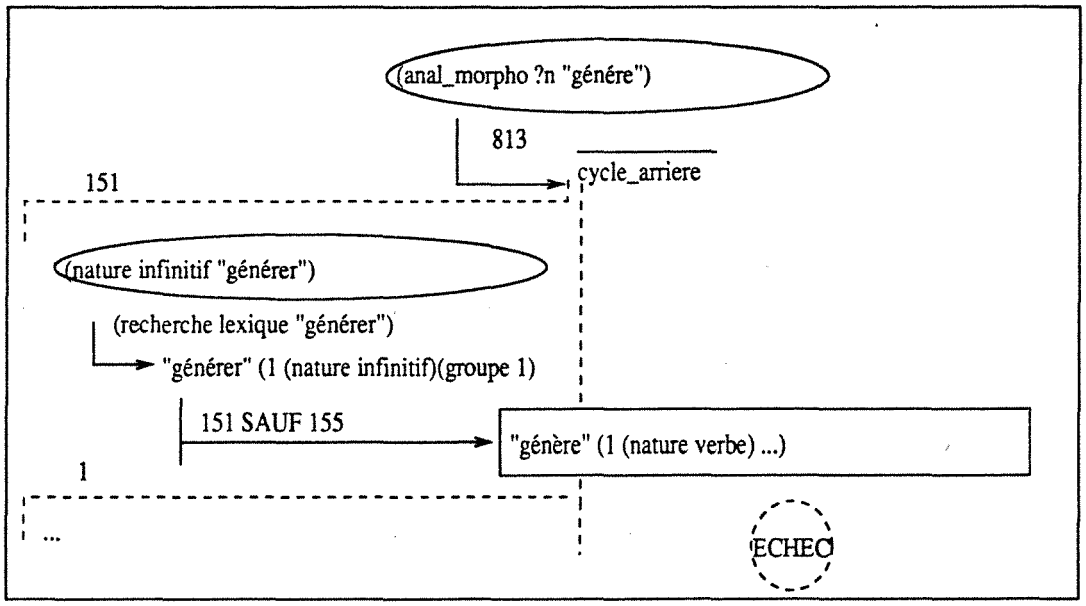


Figure 5.8. Analyse morphologique de *"génére"*.

Dans un traitement automatique, le fait de ne pas reconnaître *"génére"* est signe d'une anomalie. Un expert effectuerait rapidement le diagnostic et signalerait le non respect de la règle d'exception 155 (page 182) sur la conjugaison des verbes en *"é.rer"*. Le blocage pur et simple de l'analyse apporte cependant trop peu d'informations pour effectuer ce même diagnostic de façon assurée : est-ce que le mot recherché est bien un verbe ? doit-il être conjugué au singulier ou au pluriel ?

Une analyse tolérante pour ce type de fautes revient à considérer *"génére"* comme un verbe conjugué à la première personne du singulier du présent, malgré l'utilisation erronée des règles de conjugaison. Le succès des inférences ultérieures valide cette hypothèse et permet de mieux circonscrire la faute.

Mise en œuvre

Pour réaliser ce blocage sur les règles d'exceptions, on utilise la notion de profondeur d'une règle. Le niveau de profondeur dans le réseau des règles est incrémenté de 1 à chaque accès par le lien *Sauf*. Une règle d'entrée a le niveau 1 par défaut.

Pour limiter l'accès aux règles les plus générales, sans consulter les règles exceptions, il suffit que la condition `prof` introduite au §5.1.3 vaille : `'(> niveau 1)`.

Seules les règles de niveau 1 seront alors acessibles, puisque la condition `(> niveau 1)` est vraie pour toutes les autres. Dans l'exemple déroulé ci-dessus, on applique en particulier la règle 151, et non la règle 155 qui lui fait exception.

autre exemple :

Analyse morphologique de "floue" *:*

"floue" *est reconnu par l'application de la règle générale de construction du féminin des adjectifs terminés par une voyelle (règle 101 page 178)) sur l'adjectif* "flou". *Le déclenchement des règles exceptions pour le féminin des noms en* "-ou" *(règle 105 page 178)) ne contredit pas cette conclusion.* "flou" *est en effet une contre-exception (règle 113) de la règle 105. On aura simplement fait temporairement une économie sur les traitements effectués.*

La forme finale de la fonction `cycle_arriere_popa` devient alors :

Fonction `cycle_arriere_popa`
syntaxe : `(cycle_arriere_popa l_regles prof attente objectif)`

`l_regles` : *liste des règles à déclencher en chaînage arrière*
`prof` : *condition bloquante pour l'application des règles*
`attente` : *booléen à vrai si la stratégie appliquée doit être une stratégie en largeur d'abord sauf si une règle est immédiatement concluante*
`objectif` : *objectif à atteindre dont la réalisation provoque l'arrêt temporaire des traitements*

Détection des fautes

Avec une analyse tolérante, on a fait abstraction des règles exceptions. Elles devront cependant être contrôlées au moment de faire le bilan de l'analyse.

(810	*stratégie*	
(Si	*(objectif (impression_bilan ?n ?y)))*	*si l'objectif est d'imprimer le bilan de l'analyse sur le mot ?y*
(Et_Si	*(nature ?n ?y))*	*et si ?y est de nature ?n*
(Alors	*(vérifier_exceptions ?n ?y)))*	*alors déclencher la fonction vérifier_exceptions*

La fonction `vérifier_exceptions` déclenchée dans la règle 810 contrôle la validité des règles appliquées par rapport à leurs règles d'exceptions.

Remarques sur une stratégie tolérante aux exceptions

Cette stratégie est basée sur l'hypothèse de travail que plus les règles à suivre sont fines, plus elles ont de risque d'être mal appliquées.

- L'intérêt d'une telle démarche se retrouve dans tous les cas où l'origine d'une faute vient de l'omission d'une règle particulière. Les énoncés suivants en sont une illustration.

  "Un segment parasite est inclu_(s) dans un groupe de segments parallèles"
  où le participe passé du verbe *"inclure"* est une exception des verbes en *"-clure"*.

  "Cette ferme auberge inquiéte(ète) le jeune restaurateur du village"
  où le verbe *inquiéter* a été mal conjugué puisque la conjugaison des verbes en *"é.er"* forme une exception à la conjugaison des verbes du premier groupe.

- Le fait de retarder les vérifications permet de n'effectuer que celles qui sont nécessaires.

  On aura confirmé que *"inquiète"* est employé comme verbe et non comme adjectif dans ce dernier exemple, avant de déclencher la vérification des règles exceptions appropriées. On aura ainsi fait l'économie du déclenchement des règles plus fines sur la construction du féminin des adjectifs.

- Il faut veiller pour cela à ce que les règles inverses renvoient à la règle la plus générale qui convient. En effet, prenons le cas du mot *"barone(onne)"*. Si les règles inverses renvoient à la règle générale sur la construction du féminin des noms, il est possible :

1 - de reconnaître *"barone"* comme un nom féminin construit à partir de *"baron"* (règle 1 page 175)),

2 - de détecter ensuite l'erreur de construction par oubli de la règle exception sur les noms terminés par *"on"* (règle 6 page 177)). La forme correcte est *"baronne"*.

- Cette façon de procéder n'apporte au contraire rien quand l'erreur porte sur la transcription d'une forme irrégulière (contre-exception).

exemples :

"inquiéte(ète)" *lorsqu'il est utilisé comme adjectif. :*
Le féminin des adjectifs terminés par "t" *se forme en ajoutant un* "e" *(règle 26 page 176).*

Mais le féminin des adjectifs en "-et" *se forme habituellement en* "-ette" *(règle 107 page 180).*

Mais "inquiet" *dont le féminin est* "inquiète" *(règle 118 page 180) est une exception.*

Ni la désinence "e" *sur le radical* "inquiét", *ni la désinence* "éte" *ne permet de retrouver le radical de* "inquiéte" *et de renvoyer à une règle morphologique concluante.*

Il en est de même pour "turcque(que)".

Les méthodes à mettre en œuvre procèdent dans ce cas de la correction orthographique et se rapprochent plus d'un aspect phonétique.

5.3 Analyse syntaxique

5.3.1 Introduction

L'objectif de l'analyse syntaxique est de reconnaître si un énoncé est grammaticalement correct. Le problème majeur de l'étude d'énoncés en langue naturelle est celui du traitement des ambiguïtés qui apparaissent à tous les niveaux de l'analyse et de la multiplicité des hypothèses qui en découlent.

Nous présentons la façon d'intégrer différents types de stratégies conçues pour traiter cette question, mais auparavant nous rappelons brièvement le problème.

Les ambiguïtés rencontrées sur les constituants d'un énoncé peuvent être inhérentes à la langue :

- Un mot a souvent plusieurs usages possibles :

  exemples :

  - "la" *est utilisé dans une phrase soit comme déterminant, soit comme particule préverbale*
  - "entre" *soit comme préposition, soit comme conjugaison de* "entrer"

- Certains groupes de mots ont plusieurs interprétations syntaxiques :

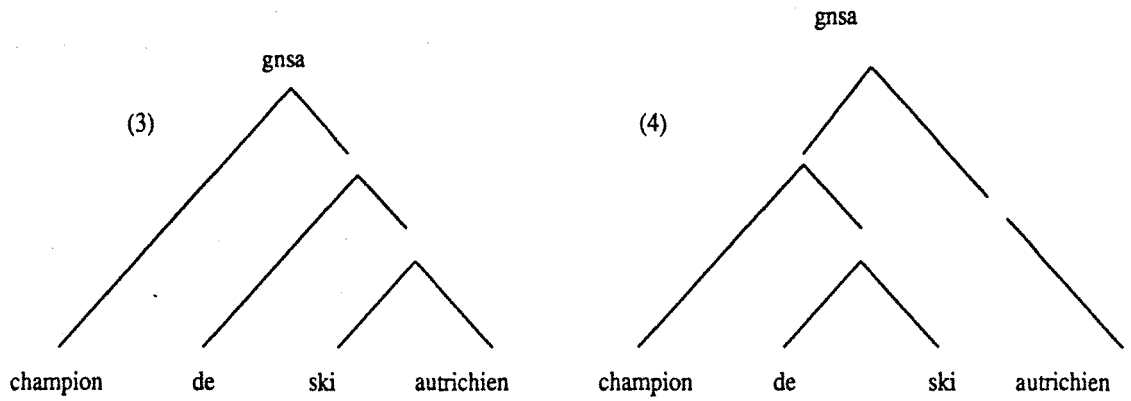
  exemple :

  Dans le groupe de mots "les frontières des îles localisées, *on ne sait pas à quel nom,* "frontières" *ou* "îles", *s'attache l'adjectif* "localisées".

Elles sont aussi souvent dues à la grammaire décrivant la langue :

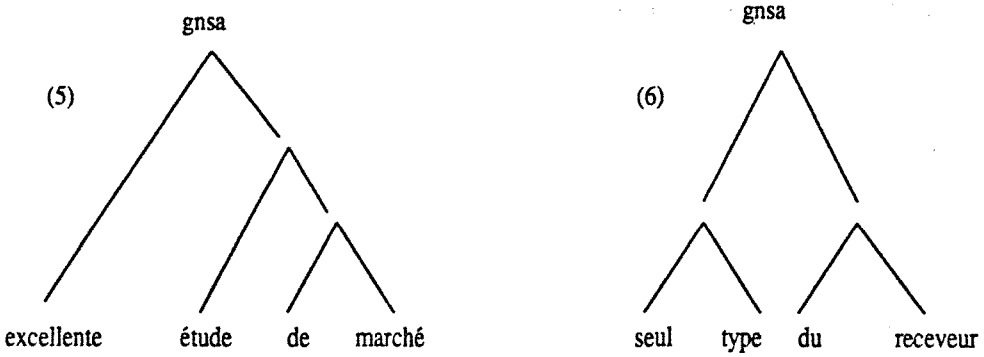
exemple : *"le champion de ski autrichien"*

Sans connaissance sémantique, une grammaire classique prévoit deux solutions, la première attache l'adjectif *"autrichien"* au nom *"ski"* , la seconde l'attache au groupe de mots *"champion de ski"*, alors que seule la seconde est acceptable eu égard au sens.



Une grammaire doit prévoir d'autant plus d'hypothèses multiples que sa couverture est grande et que des éléments de signification sont attachées aux structures syntaxiques. Ainsi pour des analyses poussées, même sans problème d'accord, il est nécessaire de faire la différence entre les structures (5) et (6) des groupes de mots :

"excellente étude du système" et "seul type du receveur"



Un système de détection de fautes d'orthographe considéré dans un cadre général comme celui des traitements de textes, doit offrir une couverture importante et prévoir ces différents types de structures syntaxiques. On se trouve donc confronté d'emblée au problème de la multiplication des hypothèses.

Certains systèmes fonctionnant en analyse gauche-droite du texte s'appuient sur des outils puissants de prédiction pour contraindre l'analyseur dans les choix des règles à appliquer [Marcus 80] et opérer une analyse déterministe. La stratégie développée par [LallichBoidin 86] consiste à exploiter les informations syntaxiques portées par les mots de la chaîne d'entrée avant de la confier à un analyseur syntaxique. En effet certains mots tels que les prépositions ou les verbes qui admettent un comportement rectionnel particulier sont porteurs d'informations syntaxiques qui complètent une grammaire hors-contexte classique. D'autres stratégies pour contraindre l'analyse syntaxique peuvent également être développées en faisant intervenir une part de sémantique dans le processus d'analyse.

Le cycle d'inférences sur les règles syntaxiques déclenché par la règle 804 (page 97) crée l'ensemble des constituants syntaxiques de l'énoncé de façon purement combinatoire, toutes les hypothèses possibles sont développées. Certaines des solutions conduiront à des impasses, les autres à des analyses pertinentes, ainsi obtiendra-t-on toutes les structures grammaticales possibles de l'énoncé. Ce traitement tend à prendre énormément de temps à énumérer et à suivre les traitements possibles, et il est intéressant de limiter tant que faire se peut l'ensemble des hypothèses de travail.

Nous présentons ci-dessous cette opération d'analyse syntaxique telle que nous l'avons abordée sur nos règles syntaxiques, dans la perspective de limiter le nombre des hypothèses

multiples qu'elle crée et de traiter des énoncés incorrects du point de vue orthographique.

Notre but n'était pas de mettre au point une nouvelle stratégie d'analyse syntaxique. Nous avons plutôt souhaité étudier l'intérêt et l'apport de la représentation des connaissances proposée pour ce problème, même si, on en conviendra, une compilation des règles syntaxiques est nécessaire pour adapter à un filtrage plus efficace les règles syntaxiques écrites comme des fonctions directement interprétables en Lisp. Nous décrivons donc quelques éléments de stratégie et la façon de les exprimer dans des règles de contrôle.

Nous pouvons déjà faire remarquer que la stratégie générale d'analyse immédiate proposée au début de ce chapitre a l'avantage de diminuer au moins dans les premières étapes le nombre d'hypothèses coexistantes. Par contre, comme nous allons le détailler, une stratégie tolérante générale augmente incontestablement le nombre de solutions parasites.

5.3.2 Limiter les solutions parasites

Nous entendons par solutions parasites les solutions qui apparaissent à un moment ou à un autre de l'analyse et qu'il serait possible d'éliminer en tenant compte des informations que l'on a sur le fonctionnement de la langue.

a) Privilégier les structures correctes

Vérification des contraintes de la clause Sous_Réserve

Les énoncés traités sont des énoncés susceptibles de contenir des fautes, la stratégie tolérante présentée dans la section 5.1.3 et qui permettait de ne pas bloquer dès la première faute est généralement trop permissive. Il est préférable de réserver cette tolérance dans les situations où elle apparait nécessaire.

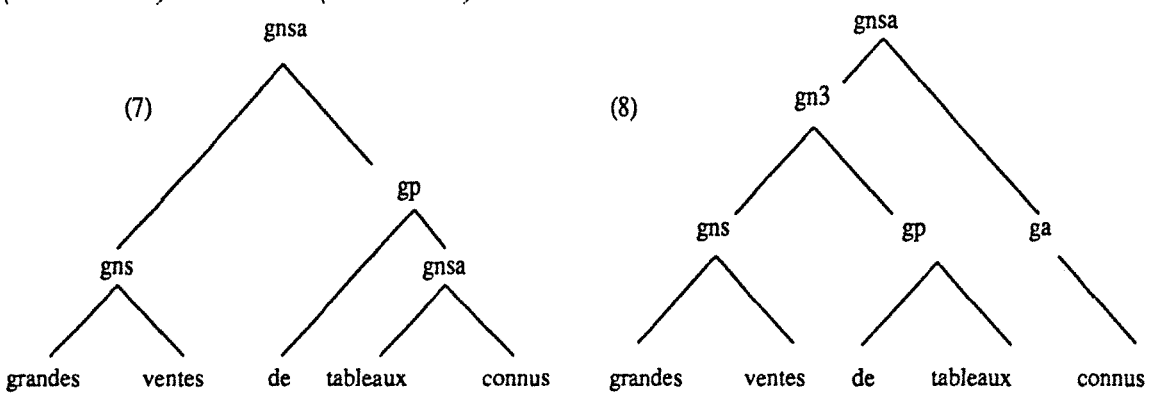
exemple 4

"Les grandes ventes de tableaux connus se déroulent à Londres"

La solution qui consiste à inhiber de façon systématique la vérification des contraintes de la clause *Sous_Réserve* (§5.1.3) ne s'avère pas toujours de bon aloi. Dans l'exemple 4, cette façon de procéder donne naissance aux deux solutions (7) et (8)[4] pour la structure du

4. Les structures ainsi représentées ne décrivent que les noeuds principaux des arbres syntaxiques.

groupe de mots *("grandes ventes de tableaux connus" 2)*[5] par la grammaire expérimentale sur laquelle nous avons travaillé. La solution (8) est une solution parasite qui serait rejetée par une analyse classique puisque l'accord en genre n'est pas respecté entre l'adjectif *("connus" 6)* et le nom *("ventes" 3)*.



Une façon d'agir est de garder cette option, et de fournir à l'utilisateur toutes les structures syntaxiques de la phrase reconnues par la grammaire. Chacune des solutions est accompagnée par un diagnostic approprié et c'est à l'auteur du document de sélectionner celle qui lui semble pertinente. Cela ne paraît pas raisonnable et peu ergonomique. En l'absence d'une analyse élaborée qui puisse déterminer la(les) solution(s) non seulement reconnues d'un point de vue grammatical mais aussi acceptables pour le sens, un autre comportement consiste à privilégier, s'il en existe, les structures correctes d'un point de vue orthographique. Plutôt que de faire cette sélection a posteriori et de procéder à un certain nombre de traitements inutiles, il est préférable de le faire au moment de l'analyse quitte à autoriser une plus grande tolérance lorsque la situation le nécessite.

Contrairement à la stratégie développée dans la section 5.1.3 nous préconisons donc, de ne pas être tolérant dans un premier temps au niveau de l'analyse syntaxique et de mettre le paramètre booléen `confirm` de la fonction `cycle_avant` à vrai de façon à obliger la vérification des conditions de la clause *Sous_Réserve*.

Comportement en cas d'erreur

exemple 5

"Cette excellentes() composition enchante les productions de jeux"

5. Ces structures portent sur les mots de la phrase, même si nous n'avons pas fait apparaître sur ces schémas les places des mots dans l'énoncé.

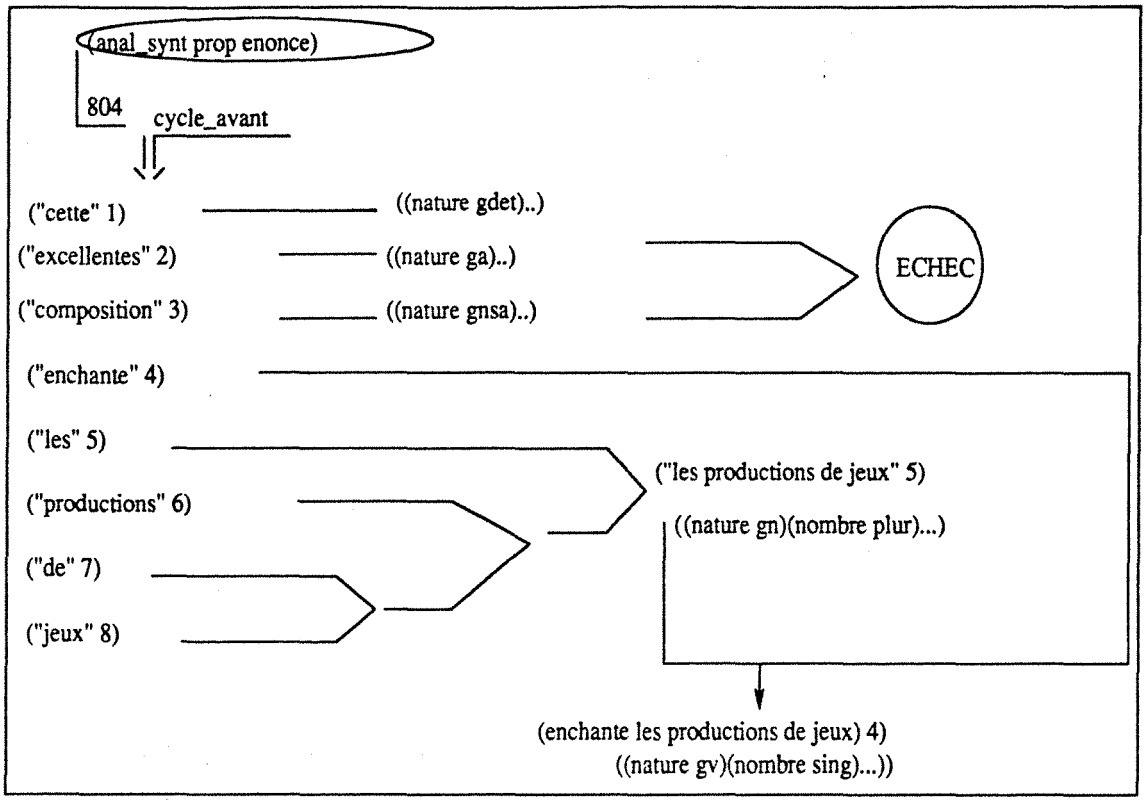


Figure 5.9. Début d'analyse de la phrase *"cette excellentes composition enchante les productions de jeux"*.

La recherche de l'objectif `(anal_synt prop enonce)` aboutit à un échec à cause de la faute d'accord entre *("excellentes" 2)* et *("composition" 3)*. Nous l'avons déjà dit, l'analyse d'un énoncé erroné est facilitée par la poursuite de l'analyse après échec, on va donc chercher à débloquer la situation.

Cela nécessite de revenir sur la décision d'analyse non tolérante et d'autoriser l'application de règles erronées et cela s'exprime par une règle de reprise (règle 841) associée à la règle de stratégie 804. Nous n'autorisons pour le moment (cf §5.1.3) qu'une impasse sur la vérification des exceptions (cf §5.2.2) et sur celle des conditions de la clause *Sous_Réserve*. Dans le cas de figure qui nous préoccupe, nous souhaitons effectuer une analyse tolérante aux fautes sur les contraintes de la clause *Sous_Réserve*. Le rôle de la règle de reprise 841 est donc de faire abstraction des conditions de cette clause pour les règles validées mais non confirmées et de forcer leur application.

Structure d'informations utilisée (cf §5.1.5) :

- `l_regles_non_confirmees`
- *fonction* (`condition_non_verifiee c r`)

(841	*reprise*	
(Si	*(objectif (anal_synt ?s enonce)))*	*si il y a échec sur l'analyse syntaxique de l'énoncé*
(Et_Si	*(distribue ?r l_regles_non_confirmees))*	*et si ?r est une règle de l_regles_non_confirmees*
(Alors	*(appliquer ?r)*	*alors appliquer ?r*
	(ajouter ?r l_regles_anomalies)))	*et ajouter ?r à l_regles_anomalies*

Pour effectuer un diagnostic des fautes rencontrées et les signaler à l'utilisateur, on est obligé de retrouver les règles entachées d'erreur. Elles sont stockées dans la liste `l_regles_anomalies`.

Remarques

- Cette stratégie s'avère efficace sur l'ensemble des phrases recueillies dans notre corpus. Elle est particulièrement agréable lorsqu'il y a présence de groupes prépositionnels imbriqués avec adjectifs, ou quand il y a des propositions relatives.

  exemples :

  "Ce jeune chef d'entreprise rémois part à la conquête(être) de l' Amérique"

  ("rémois" 6) *est pris comme adjectif qualificatif de* ("chef" 3) *et non de* ("entreprise" 5).

  "Un segment parasite est inclu_(s) dans un groupe de segments parallèles"

  "parallèles" *est attaché à* "segments" *et non à* "groupe".

- Lorsqu'il existe une faute sur une structure ambiguë, cette stratégie peut être mise en défaut :

  "Cet article portait sur le comportement d'une variable hérité_(e)"

Par la précédente stratégie, le syntagme *("comportement d'une variable hérité" 6)* est reconnu en supposant que *("hérité" 10)* qualifie *("comportement" 6)* alors qu'il est attaché réellement à *("variable" 9)*.

Dans cet exemple, une analyse syntaxique plus fine s'appuyant sur les compléments régis par le *"comportement"* permettrait éventuellement de retrouver la structure correcte [LallichBoidin 86]. Il n'en est pas de même pour tous les énoncés. On a en effet constaté pour la plupart des exemples relevés dans notre corpus, que la levée de l'ambiguïté ne pouvait être effectuée qu'à un niveau sémantique. Hors d'un domaine de travail bien circonscrit, il est difficile d'introduire des informations à ce niveau, et nous avions posé comme hypothèse de travail que nous ne le ferions pas.

exemples :

"Il a l'impression de travailler sur un éditeur de textes classiques()"

"Le problème de reconnaissance d'objets se ramène à une mise en correspondance entre une certaine représentation de l'image et une représentation des objets connu_(?) par le système"

- Du fait de l'ambiguïté de la grammaire, la liste `l_regles_non_confirmées` est rarement réduite à une seule règle. La règle de reprise 841 applique donc successivement toutes les règles validées mais non confirmées jusqu'à ce qu'une d'entre elles permette de débloquer l'analyse. Dans un cas extrême, on déclenche l'application de toutes ces règles et on atteint le même résultat qu'avec une analyse tolérante. Pour limiter ce risque, il faut envisager des *stratégies de reprise* afin de repérer les situations les plus suspectes et d'orienter la reprise vers leur "redressement". La section 5.4 présente des exemples de stratégies pour ce faire.

Explicitation des erreurs

L'explicitation des erreurs revient en cas de stratégie d'analyse syntaxique non tolérante avec reprise, à afficher les règles de la liste `l_regles_anomalies` utilisées pour construire la structure de la phrase. Ou bien, quand cette structure n'a pas pu ête déterminée, à afficher toutes les règles de `l_regles_anomalies`.

(811	*strategie*	
(Si	*(objectif (impression_bilan ?s enonce)))*	*si l'objectif est de faire le bilan de l'analyse de l'énoncé*
(Et_Si	*(non (nature ?s enonce)))*	*et si la structure ?s n'a pas pu être reconnue pour* **enonce**
(Alors	*(ecrire "structure de l'énoncé non reconnue")*	
	(ecrire " mais on peut dire que ")	
	(r_a (impression_anomalies_accord))))	*imprimer les erreurs sur les accords*

(818	*stratégie*	
(Si	*(objectif (impression_anomalies_accord)))*	*si l'objectif est d'imprimer les anomalies d'accord rencontrées*
(Et_Si	*(distribue ?r l_regles_anomalies)*	*et si ?r est une règle de l_regles_anomalies*
	(condition_non_verifiee ?c ?r))	*et que ?c est la condition non vérifiée de ?r*
(Alors	*(ecrire "la condition" ?c "de la règle" ?r "n'a pas été respectée ")))*	

(819	*strategie*	
(Si	*(objectif (impression_anomales_accord)))*	*si l'objectif est de faire le bilan de l'analyse de l'énoncé*
(Et_Si	*(nature ?s enonce)*	*et si l'énoncé a la structure ?s*
	(distribue ?r l_regles_non_confirmees)	*et si ?r est une règle de l_regles_anomalies*
	(est_dans_renvoi ?r ?s enonce)	*et si ?r a été utilisée pour définir la structure ?s de énoncé*
	(condition_non_verifiee ?c ?r))	*et que ?c est la condition non vérifiée de ?r*
(Alors	*(ecrire "la condition" ?c "de la règle" ?r (" n'a pas été vérifiée")))*	

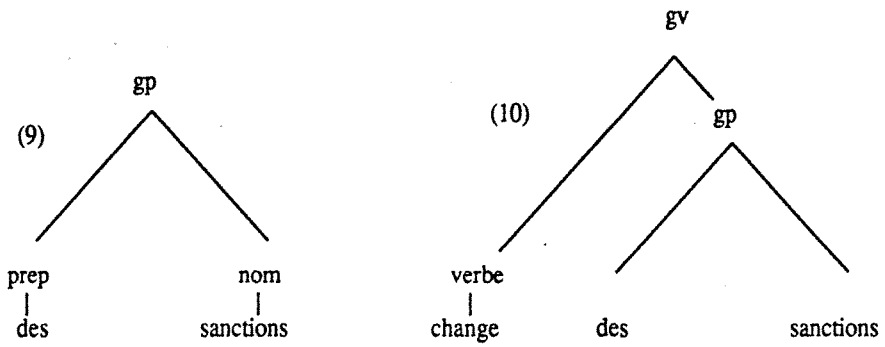
b) Supprimer certaines ambiguïtés au niveau du mot

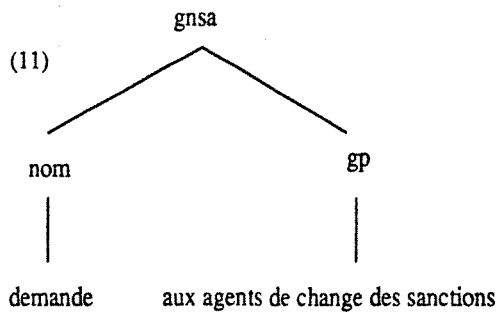
La stratégie d'analyse immédiate limite de façon arbitraire le nombre d'hypothèses à prendre en compte sur un mot. Elle peut être complétée par des stratégie s'appuyant sur les connaissances linguistiques disponibles. En effet, des prétraitements sur les règles décrivant la langue permettent d'extraire des informations pertinentes exploitables par les stratégies.

exemple 6

"La société demande aux agents de change des sanctions"

Les ambiguïtés de départ sur les mots *"change"* , nom ou verbe, et *"des"* , préposition ou article, créent les types de solutions parasites (9) et (10). Celle sur *"demande"*, nom ou verbe, crée la solution parasite (11).





Plutôt que d'abandonner une solution parce qu'elle ne satisfait pas à la reconnaissance de la phrase, on peut tenter de lever le plus tôt possible certaines ambiguïtés en effectuant une étude préliminaire du contexte d'utilisation du mot dans la phrase. Des informations complémentaires sur la langue ou une compilation adéquate des connaissances du système permettent parfois d'éliminer des solutions parasites, incompatibles avec le contexte immédiat du mot dans la phrase. Par exemple, [Pierrel 81], le groupe Sydo [Antoniadis 84] et [Charpillet 85] utilisent des règles contextuelles, [Sedogbo 83] met en œuvre des matrices de précédence. [Catach 84a] a également défini un certain nombre de règles de désambiguïsation utilisées dans un système de phonétisation automatique. Il est possible également de faire intervenir des informations statistiques sur le corpus traité notamment sur les biclasses ou les triclasses [Fluhr 77].

Nous illustrons ci-dessous la mise en place de telles démarches par une stratégie s'appuyant sur la compatibilité entre voisins grammaticaux.

Le principe de cette stratégie est de tester, avant de propager à un mot de la phrase une hypothèse sur le mot de la langue associé, si cette hypothèse est compatible avec celles sur ses voisins immédiats dans la phrase.

Pour pouvoir traiter chaque mot individuellement, la règle 815 ne devient qu'un intermédiaire pour lancer la reconnaissance de chacun des mots de la phrase.

(815	*stratégie*	
(Si	*(objectif (definir ?l)))*	*si l'objectif est de définir les éléments de la liste ?l*
(Et_Si	*(distribue ?x ?l))*	*et si ?x est un élément de ?l*
(Alors	*(r_a (definir_mot ?x))))*	*alors chercher à définir ?x*

(816	*stratégie*	
(Si	*(objectif (definir_mot ?x)))*	*si l'objectif est de définir le mot ?x*
(Et_Si	*(chaîne ?y ?x)*	*et si ?y est le mot de la langue associé à ?x*
(Alors	*(r_a (nature ?n ?y))*	*alors chercher auparavant une hypothèse grammaticale sur ?y*
	(r_a (propager_hyp ?y ?x))))	*et chercher à propager les hypothèses sur ?y à ?x*

(817	*stratégie*	
(Si	*(objectif (propager_hyp ?y ?x)))*	*si l'objectif est de propager les hypothèses de ?y à ?x*
(Et_Si	*(nature ?n ?y)*	*et si ?y peut avoir pour nature ?n*
	(compatible_voisin ?x ?n))	*et que ?n est compatible avec les hypothèses sur les voisins de ?x*
(Alors	*(attribuer (nature ?n ?y) ?x)))*	*alors attribuer cette hypothèse à ?x*

La trace suivante donne un extrait de l'analyse de l'exemple 6, sachant que :

- la stratégie générale est une stratégie d'analyse immédiate ;

- l'analyse morphologique est effectuée avec ordre de désinence décroissante et priorité aux hypothèses dans l'ordre arbitraire nom, adjectif, verbe ;

- dans la grammaire utilisée, une suite de deux noms n'est pas autorisée ce qui rend incompatible l'hypothèse *(("demande" 3) ((nature nom)...))* avec l'hypothèse *(("société"*

té" 2)((nature nom) ...)).

On peut remarquer que si les hypothèses sur les verbes avaient été favorisées par rapport aux règles sur les noms, la reprise à effectuer aurait porté sur *"change"* et non sur *"demande"*.

Remarques

- Les ambiguïtés potentielles portent principalement sur des mots qui peuvent être nom ou adjectif, ou bien nom ou verbe. Nous avons remarqué que cette stratégie apporte peu si la grammaire de la langue accepte une suite de deux noms. Dans ce cas en effet, si on se limite à la notion de voisin immédiat, de nombreuses solutions restent acceptables au niveau local (cf exemple précédent).

- La notion de voisin immédiat ne suffit donc pas toujours. Elle nécessiterait parfois d'être agrandie aux voisins des voisins, pour lever par exemple les ambiguïtés sur *("dernières" 2)* nom ou adjectif et *("analyse" 3)* nom ou verbe dans l'exemple 7.

  exemple 7
  "cette dernière analyse modifie le marché américain"

5.3.3 Exploiter la hiérarchie des règles

Sans stratégie pour seconder l'analyseur syntaxique, le nombre des hypothèses multiples apportées par l'ambiguïté de la grammaire ne peut se réduire qu'en favorisant de façon arbitraire certaines des solutions par rapport aux autres.

Avec la structure adoptée pour les règles syntaxiques, les ambiguïtés apparaissent au niveau des règles liées par un lien *Ou*.

exemple 8
"On prend un segment connecté à un des bouts de segments étiquetés"

Soient les règles 515 (page 203), 509 (page 203) et 505 (page 202) présentées sous forme condensée ci-après :

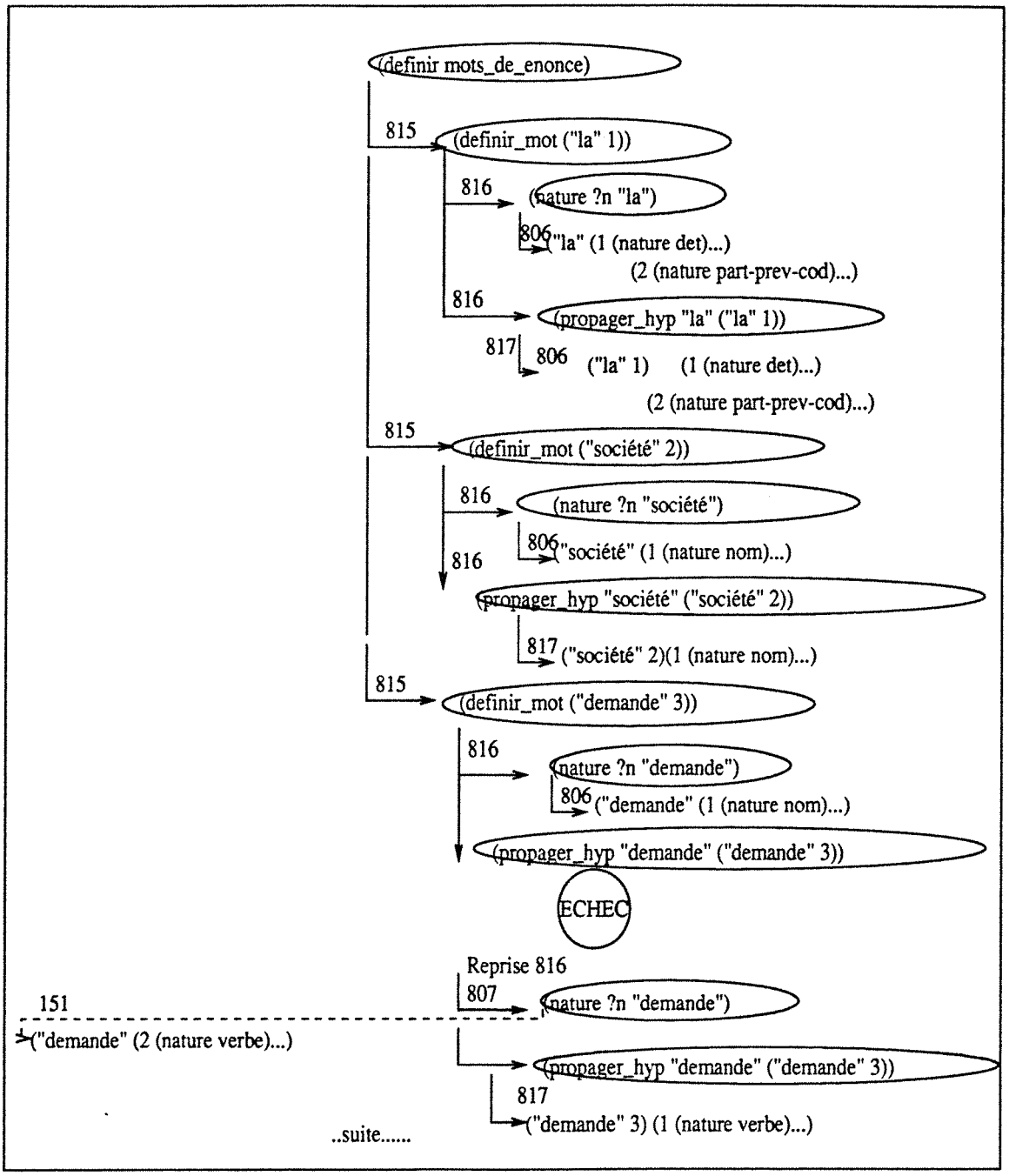


Figure 5.10. Extrait de la recherche d'hypothèses grammaticales sur les mots de la phrase *"la société demande aux agents de change des sanctions"*.

515 : GN3 ← GNS Ou 514

514 : GN3 ← GNS GP

509 : GN4 → GN3 Ou 513, 560

560 : GN4 ← GN3 GA

505 : GNSA ← GN4 Ou 512

512 : GNSA ← GN4 GP

L'analyse syntaxique du segment de phrase *("bouts de segments étiquetés" 9)* peut se faire par les deux chemins décrits dans les figures 5.11 et 5.12.

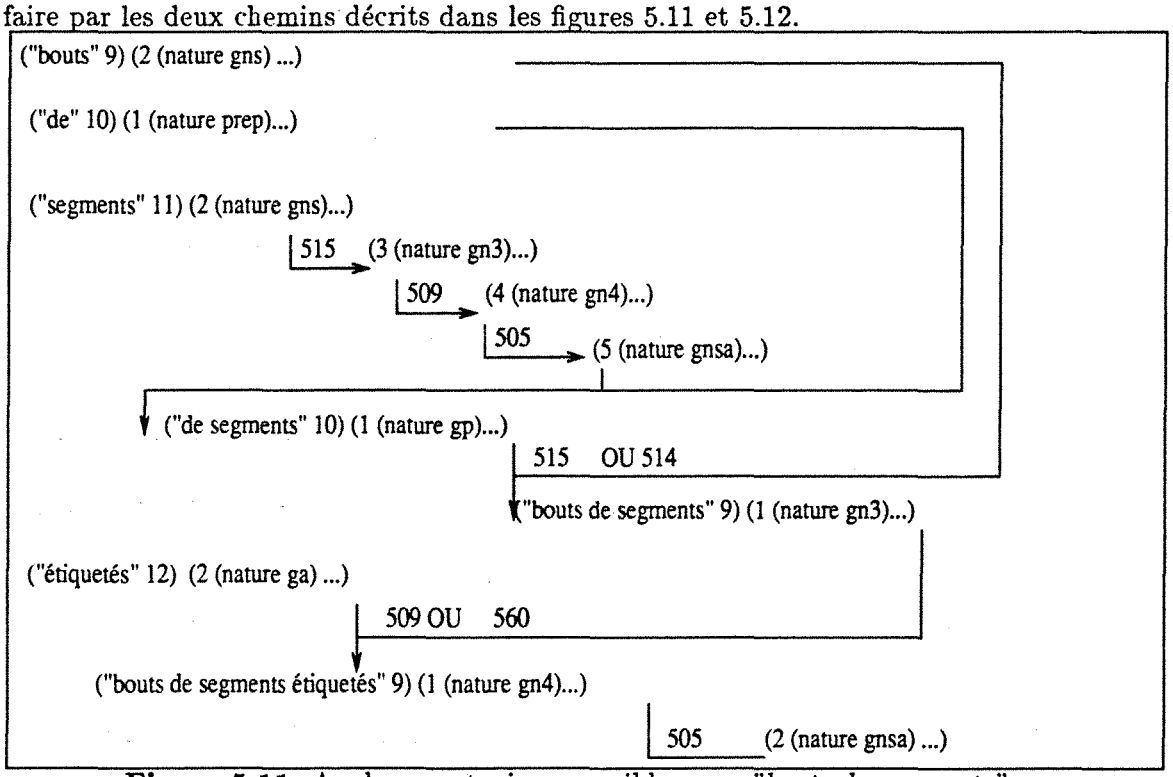


Figure 5.11. Analyse syntaxique possible pour *"bouts de segments"*.

Afin de limiter les solutions inférées par les règles syntaxiques, il est intéressant lorsqu'une règle d'entrée et une de ses règles *Ou* sont concurrentes, d'en favoriser une des deux.

Vu la façon dont nous avons construit les règles syntaxiques, le fait de favoriser les règles *Ou* revient à favoriser les règles qui proposent à un niveau local, un recouvrement maximum du texte.

La mise en œuvre d'une telle stratégie passe par l'intermédiaire du paramètre `"priorite_ou"` de la fonction `cycle_avant` dont nous pouvons maintenant donner la syntaxe générale :

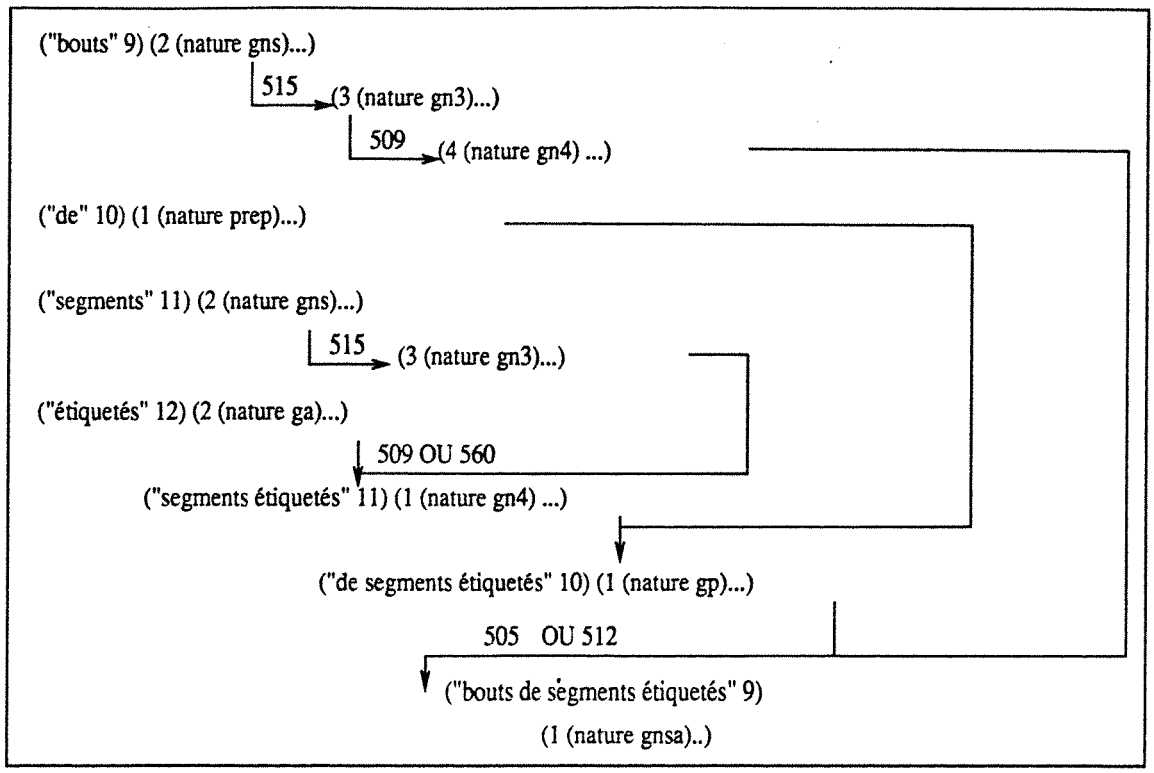


Figure 5.12. Analyse syntaxique possible pour *"bouts de segments"*.

fonction `cycle_avant`

syntaxe : `(cycle_avant l_regles enonce st confirm prof priorite_ou)`

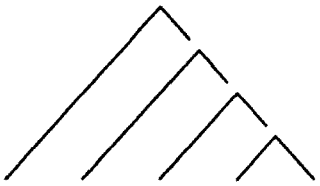
`l_regles`		liste des règles sur lesquelles effectuer le cycle d'inférences
`enonce`		enonce à analyser
`st`		structure grammaticale recherchée pour `enonce`
`confirm`	booléen	à vrai quand il est demandé une vérification des contraintes de la clause *Sous_Réserve* à faux sinon
`prof`		condition de blocage pour les règles à appliquer
`priorite_ou`	= "ou"	quand il y a ambiguïté sur l'application d'une règle d'entrée et d'une de ses règles *OU*, on donne priorité à la règle *OU*
	= "gen"	quand il y a ambiguïté sur l'application d'une règle d'entrée et d'une de ses règles *OU*, on donne priorité à l'application de la règle d'entrée
	= ()	on applique toutes les règles applicables

- Lorsque le paramètre `priorite_ou` est égal à `"ou"`, le cycle d'inférences fonctionne de telle façon qu'une règle n'est appliquée que si on sait qu'aucune de ses règles *Ou* ne peut l'être. Pour cela, il faut, avant la validation d'une règle, avoir inféré toutes les hypothèses qui pourraient influer sur celle-ci. A chaque cycle d'inférences, les règles syntaxiques sont donc ordonnées de telle façon qu'une règle dont le résultat peut servir à la validation d'une autre règle est rangée avant.

  Par exemple, il faut avoir déduit que *"étiquetés"* est un groupe adjectival (*GA*) avant de pouvoir choisir d'appliquer la règle 509 ou la règle 515 sur (*"segments" 11*) et (*"étiquetés" 12*).

  Le fait d'appliquer les règles accessibles par le lien *Ou* (seconde solution sur l'exemple 8) et de ne pas appliquer les règles plus générales revient à favoriser les structures syntaxiques que nous appelerons régulières, c'est à dire les groupes à structures de "peigne".

exemple :



- Quand le paramètre `confirm` est actif, il est plus fort que le paramètre `priorite_ou`, c'est à dire que si `priorite_ou` vaut `"ou"`, à une règle *Ou* validée mais non confirmée, on préfère sa règle d'entrée validée et confirmée.

La forme exacte de la règle 804 devient :

(804 stratégie	
(Si (objectif (anal_synt ?n enonce)))	*si l'objectif est l'analyse syntaxique de l'énoncé*
(Alors (r_a (définir mots_de_enonce))	*alors chercher à définir auparavant chacun des mots de l'énoncé*
(cycle_avant l_regles_synt ?n enonce confirm prof priorite_ou)))	*et lancer un cycle d'inférences sur les règles syntaxiques*

5.4 Stratégies de reprise

Les règles de reprise sont associées à des règles de stratégie, elles sont appliquées dans l'odre où elles sont données lorsque l'objectif recherché n'a pas été atteint par la stratégie.

Dans les règles de reprise décrites ci-dessus, nous avons présenté différentes actions de reprise associées aux opérations déclenchées. Cependant, aucune véritable stratégie n'a été décrite pour orienter la reprise vers des actions semblant plus appropriées que d'autres. Nous montrons ici qu'il est possible de guider la reprise en tentant de localiser les points d'embarras et en fournissant pour cela quelques critères simples pour repérer les situations "suspectes".

5.4.1 Etude des structures partielles reconnues

Une façon de procéder consiste à essayer d'avoir une vue d'ensemble de l'état de l'analyse au moment du blocage et d'en repérer des situations pertinentes.

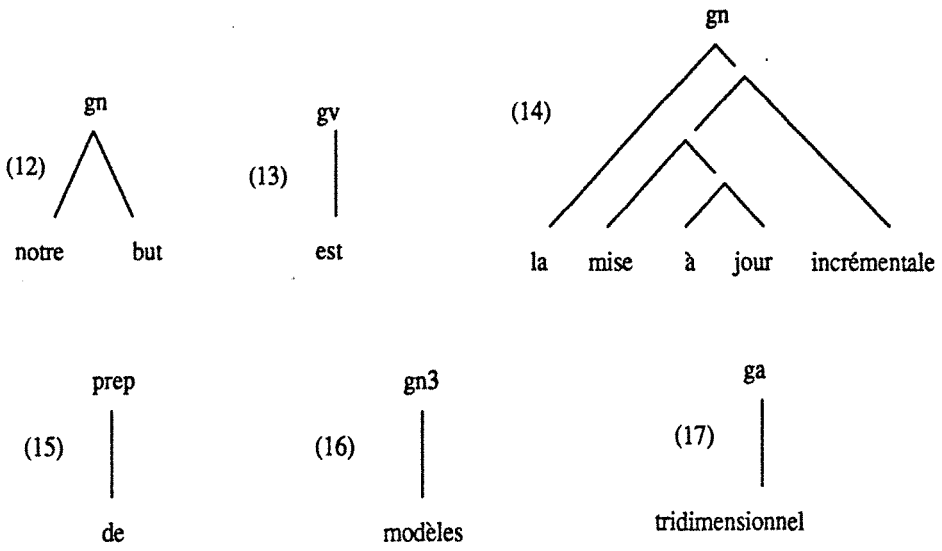
exemple 9

"notre but est la mise à jour incrémentale de modèles tridimensionnel_(s)"

L'analyse cette phrase échoue sur la faute d'accord en nombre entre le nom (*"modèles" 10*) et le nom (*"tridimensionnel" 11*). Au moment de l'étape de reprise sur l'objectif `(anal_synt prop enonce)`, si on veut effectuer une récupération des règles non confirmées (cf §5.3.2), la liste de ces règles n'est pas réduite à un seul élément :

```
L_regles_non_confirmees =
      ((560 (("jour" 7).2)(("incrémentale" 8).2))
            (genre fem ("incrémentale" 8))
       (560 ((("modèles" 10).3)(("tridimensionnel" 11).2)))
            (nombre plur ("tridimensionnel" 11)))
```

les structures partielles offrant un recouvrement maximum du texte obtenues à cette étape de l'analyse sont les suivantes :



Ce que l'on cherche dans l'étape de reprise c'est obtenir des informations qui vont faire redémarrer l'analyse. Pour cela, on préfère appliquer une règle qui construit un nouveau constituant syntaxique plutôt qu'une règle qui définit une autre structure à un constituant déjà reconnu. C'est pourquoi on souhaite favoriser la seconde règle de `l_regles_non_confirmées` qui opère sur les constituants isolés (16) et (17), c'est à dire sur des constituants qui ne sont encore intégrés dans aucun autre constituant plus grand.

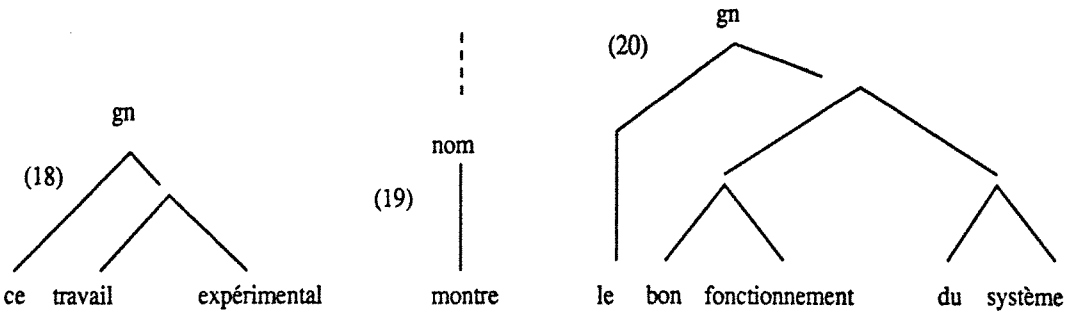
Cette démarche peut également être suivie dans d'autres types de reprise. Par exemple, dans le cas d'une stratégie d'analyse immédiate, l'objectif `(anal_synt ?s enonce)` peut échouer et nécessiter de chercher de nouvelles hypothèses sur les mots de la phrase. Plutôt que de remettre en cause les hypothèses obtenues sur chacun des mots, il est intéressant de repérer ceux sur lesquels l'analyse paraît être bloquée.

exemple 10

"ce travail expérimental montre le bon fonctionnement du système"

Les structures syntaxiques décrites ci-dessous représentent l'état de la base d'hypothèses au moment du blocage de l'analyse. On peut orienter en priorité la reprise vers la recherche de nouvelles hypothèses sur *"montre"* qui est isolé parmi les informations obtenues.

Les règles de reprise 844 et 845 décrivent la mise en place de ces stratégies.



(844	*reprise*	
(si	*(objectif (anal_synt ?s enonce)))*	*si l'objectif est d'effectuer une analyse syntaxique de l'énoncé*
(Et_Si	*(distribue Lregles_non_confirmees ?r)*	*et si ?r est une règle validée mais non confirmée*
	(est_isole ?x)	*et que ?x est un constituant isolé*
	(dans_contexte ?x ?r))	*et que ?x apparaît dans le contexte de validation de ?r*
(alors	*(appliquer ?r)*	*alors appliquer ?r*
	(complete Lregles_non_confirmees ?r))	*compléter la liste des règles appliquées malgré la présence d'une anomalie avec ?r*

(845	*reprise*	
(si	*(objectif (anal_synt ?s enonce)))*	*si l'objectif est de définir les mots de la liste ?l*
(et_si	*(distribue ?x mots_de_enonce)*	*et si ?x est un mot de l'énoncé*
	(est_isole ?x))	*et que ?x est un mot isolé*
(alors	*(reprendre_objectif (definir_mot ?x))))*	*alors reprendre la recherche d'hypothèses grammaticales sur ?x*

Cette règle utilise la fonction est_isolé qui repère les constituants isolés parmi les structures syntaxiques inférées.

Une autre idée de stratégie est de s'aider d'une analyse descendante s'appuyant sur les syntagmes déjà reconnus pour repérer les points d'embarras. Cette stratégie nécessite cependant des traitements plus conséquents. L'avantage de la stratégie de reprise présentée ci-dessus est qu'elle est peu coûteuse et facile à mettre en œuvre.

5.4.2 Priorité aux règles exceptions

La seconde stratégie de reprise que nous présentons est moins générale que la précédente. Elle n'apparaît en effet intéressante que pour la reprise sur les règles non confirmées. Elle s'appuie sur la hiérarchie qui existe entre les règles.

exemple 11

"cette excellentes() composition enchante les productions de jeux télévisés"

Cet exemple qui reprend l'exemple 5 diffère par la présence de l'adjectif (*"télévisés" 9*). L'analyse de cette phrase est similaire à celle décrite succintement en figure 5.9, mais au moment de l'étape de reprise, la liste des règles non confirmées n'est plus réduite à un seul élément :

```
L_regles_non_confirmees =
((517 (("excellentes" 2) . 2)(("composition" 3) . 1))
      (nombre sing ("excellentes" 2))
 (560 (("productions de jeux" 6) . 1)(("télévisés" . 9) . 2))
      (genre fem ("télévisés" . 9)))
```

On remarque que la règle 517 (page 204) est une règle exception de la règle 516 (page 203) :

516 : GNS → NOM Sauf 517

517 : GNS → ADJ NOM

sous réserve de l'accord en genre et en nombre entre l'adjectif et le nom

La règle 517 est filtrée et validée, elle aurait dû empêcher l'application de la règle 516 et être appliquée en priorité. L'aspect *obligatoire* des règles exceptions incite en cas de blocage à leur imputer l'erreur et à forcer leur application même en présence d'une anomalie.

Structure d'information utilisée :

`l_regles_sauf` : liste des règles qui ont un statut d'exception.

Règle

La règle de reprise 846 qui met en œuvre cette stratégie est donnée ci-dessous.

(846	*reprise*	
(si	*(objectif (anal_synt ?p enonce)))*	*si l'objectif est l'analyse syntaxique de l'énoncé*
(et_si	*(distribue Lregles_non_confirmees ?r)*	*et si ?r est une règle validée mais non confirmée*
	(distribue Lregles_sauf ?r))	*et que ?r est également une règle exception*
(alors	*(appliquer ?r)*	*alors appliquer ?r*
	(complete Lregles_anomalies ?r)))	*compléter avec ?r la liste des règles appliquées malgré la présence d'une anomalie*

Cette stratégie est intéressante pour toutes les phrases dans lesquelles la faute est localisée sur un adjectif placé avant le nom (règle 516 page 203 sauf 517 page 204) et pour les phrases avec une erreur d'accord entre le sujet et l'attribut du sujet (règle 501 sauf 502 page 201).

exemples :

"ce sont de fausse_(s) pierres de polystyrène collées sur une armature"

"la bonne méthode est recherché_(e) en fonction du seul type du receveur"

Il existe des situations où aucune des règles de reprise données ci-dessus ne peut s'appliquer. Il faut donc prévoir de laisser les règles de reprise générales et de les combiner avec les stratégies plus spécifiques, par exemple associée à la règle 804, la règle de reprise 841 après la règle de reprise 844 : (804 841 844).

5.5 Extraits et analyses de résultats

Dans les sections précédentes, nous avons montré l'intérêt des stratégies de façon locale, sur des phrases pour lesquelles elles étaient appropriées. Il faut voir maintenant si une stratégie générale commune peut satisfaire à l'ensemble des énoncés ou tout du moins à une grande majorité d'entre eux, et dans ce cas les "mauvais" effets qu'elle a sur les autres.

Nous avons extrait de l'ensemble des tests effectués sur notre corpus, les résultats obtenus sur 16 phrases analysées avec différentes stratégies.

Les stratégies mises en œuvre opèrent sur notre représentation des connaissances et nous avons principalement étudié les problèmes de reprise sur faute ou sur analyse im-

médiate. Ces résultats sont à interpréter relativement à la grammaire expérimentale que nous avons écrite.

A travers les phrases étudiées, nous avons essayé de représenter l'ensemble des énoncés de notre corpus, par les différents types de fautes qu'elles contiennent et par le contexte dans lequel ces fautes apparaissent.

Nous avons procédé de telle façon que chaque stratégie est présentée par rapport à la précédente et fait intervenir un ou deux éléments différents par rapport à cette dernière.

Ensemble des phrases analysées

1. "L' environnement de conception génére(ère) un certain nombre de directives"
2. "Cette excellentes() composition enchante les productions de jeux télévisés"
3. "Le graphe est un ensemble de segments de droite obtenus par approximation polygonales() des contours"
4. "On prend un segment connecté à un des bout_(s) de la chaîne de segments étiquetés"
5. "Ces dernières constitue(ent) l'ensemble des équations sémantiques"
6. "La bonne méthode est recherché_(e) en fonction du seul type du receveur"
7. "Nous avons déjà définis() des méthodes pendant la création des classes"
8. "Le modèle contient différentes frontières d' îles représentés(es) de la même manière"
9. "Les sections qui suivent décrivent de manière détaillée le_(s) fonctionnalités offertes par le système aux utilisateurs des environnements générés"
10. "Le nombre d'indices image_(s?) est généralement important"
11. "Les résultats expérimentaux montre(ent) que le système fonctionne bien dans des situation_(s) de recouvrement partiel"
12. "La société des agents de change demande des sanctions"
13. "Les descriptions des modules son_(t) dépendantes de la syntaxe des langage_(s)"
14. "Le principe des fonctions génériques est extrè(ê)mement sympathique"
15. "Ces contraintes que nous _(ne) supposerons que binaires créent donc un réseau d'arcs sur ces points"

16. "Cet article portait sur le comportement d'une variable hérité_(e)"

5.5.1 Effets de la tolérance

Stratégie tolérante

Les avantages et inconvénients d'une stratégie tolérante se retrouvent assez bien de façon intuitive et sont confortés par les résultats présentés sur les 6 phrases : phrase 1, phrase 3, phrase 5, phrase 11, phrase 13 et phrase 16.

L'analyse tolérante suivie ici est une analyse multiple pour que l'on puisse se rendre compte de ses effets sur la création de solutions parasites.

Nous allons donc donner pour ces 6 phrases le nombre de solutions parasites générées à cause de la tolérance autorisée au niveau syntaxique. Les comparaisons se font par rapport à la phrase corrigée.

Type de stratégie : **Complète et tolérante**
Base des règles ((800)(801)(802)(803)(804)(805)(809)(810)(811)(812)(813))
Confirm = faux
Prof = '(> niveau 1)
Priorite_ou = ()

Phrase	*Détection*	*Surdétection*	*Solutions parasites engendrées par la tolérance*	
1	oui	non	0	
3	oui	oui	10	***2**
5	oui	oui	8	accord *"ensemble"* et *"sémantiques"* accord *"ces dernières"* et *"constitue"* à la 1ère personne
11	oui	oui	2	accord *"situation"* et *"partiel"* accord *"le système"* et *"fonctionnne"* à la 1ère personne
13	1/2	oui		***1 *3**
16	oui	oui	1	accord *"comportement"* et ?*"hérité"*

Les chiffres précédés d'une étoile renvoient à des commentaires ci-dessous.

***1** La faute d'accord sur le nom *"langages"* est diagnostiquée par la signalisation de toutes les règles non confirmées. La seconde erreur est décelée du fait de l'échec sur la reconnaissance de la structure de la phrase, mais elle n'est pas du tout localisée sur la forme *"son"*.

On retrouve le même type de situation pour l'analyse des phrases 10 et 15, même si la cause de l'échec est imputable pour la première à une faiblesse de la grammaire qui n'accepte pas une suite de deux noms et pour l'autre à une faute du rédacteur.

***2** Le nombre important de solutions parasites provient de la combinaison entre les solutions multiples dues à l'ambiguïté de la grammaire et les solutions autorisées par une analyse tolérante.

***3** La surdétection vient du fait que *("des langages" 10)* peut être reconnu comme groupe nominal ou groupe prépositionnel. On peut éviter ce genre d'inconvénients en affinant l'explication des erreurs pour qu'un diagnostic similaire sur un même constituant ne soit pas répété.

Pour comparer avec la stratégie inverse, nous présentons sur ces mêmes phrases les résultats obtenus avec une stratégie non tolérante au niveau syntaxique.

Stratégie non tolérante

Nous ajoutons donc aux règles de stratégies la règle de reprise 841 (cf page 125) sur les règles non confirmées. La tolérance au niveau morphologique est au contraire toujours admise.

Type de stratégie	: **Complète, tolérante au niveau morphologique non tolérante au niveau syntaxique**
Base des règles	((800)(801)(802)(803)(813)(804 841)(805)(809) (810)(811)(818)(819))
Confirm	= vrai
Prof règles morphologiques	= (> niveau 1)
Prof règles syntaxiques	= ()
Priorite_ou	= ()

Nous regardons sur ces exemples si les erreurs sont toujours repérées ainsi que le nombre de reprise que l'aboutissement de l'analyse nécessite.

Phrase	*Détection des fautes*	*Surdétection*	*Reprises*	
1	oui	non	0	
3	oui	non	5	***4**
5	oui	non	8	***4**
11	oui	non	8	***4**
13	1/2	non	8	***1 *4**
16	non	-	0	***5**

***4** L'ensemble des reprises proposées vient du développement d'une analyse complète et de la création de toutes les solutions partielles possibles telles que :

"droites obtenus"

et *"droites obtenus par approximation polygonales"*

et *"segments de droites obtenus"* etc...

Les reprises sont effectuées jusqu'à ce qu'une d'entre elles permettent de débloquer l'analyse.

***5** Sur cette phrase, la stratégie d'analyse syntaxique est tenue en échec. En effet, par suite de la stratégies d'analyse non tolérante la structure sans faute qui attache *"hérité"* à *"comportement"* et non à *"variable"* est préférée à la structure avec faute qui correspond pourtant à la bonne solution.

Remarques générales

1) Le nombre d'hypothèses multiples, pénalisant pour l'analyse syntaxique et pour les reprises sur erreurs, reste important malgré la non tolérance. On voudrait pallier cet inconvénient :

- avec une stratégie immédiate ;
- avec une stratégie qui privilégie les règles *Ou* lorsqu'il y a ambiguïté.

2) Les reprises effectuées avec la règle de stratégie 841 ne sont pas contrôlées et dépendent de l'ordre de rangement des règles dans la liste des règles non confirmées. Plutôt que de prendre arbirairement les règles les unes à la suite des autres dans cette liste, on aimerait guider la reprise vers celles qui paraissent les plus adéquates.

5.5.2 Privilégier le lien *Ou*

Pour étudier l'effet du lien *Ou*, nous allons analyser les résultats des analyses des phrases avec le paramètre `priorite_ou` à `"ou"` pour contrôler son effet sur la détection des fautes et sur les étapes de reprise.

Type de stratégie	: **Complète, tolérante au niveau morphologique non tolérante au niveau syntaxique analyse syntaxique avec priorité aux règles *Ou***
Base des règles	((800)(801)(802)(803)(813)(804 841)(815)(816)(817)(809)(810) (811)(818)(819))
Confirm	= vrai
Prof règles morphologiques	= (¿ niveau 1)
Prof règle syntaxique	= ()
Priorite_ou	= "ou"

Phrase	*Détection des fautes*	*Surdétection*	*Reprises*	*Reprises sur faute*	
1	oui	non	0	0	
2	oui	non	1	1	
3	oui	non	2	1	
4	oui	oui	1	1	***6 *8**
5	oui	oui	1	1	
6	oui	oui	1	1	
7	oui	oui	1	1	***6**
8	oui	non	1	1	
9	oui	non	1	1	
10	non	non	0	0	***9**
11	oui	oui	8	1	***7**
12	-	-	0	0	
13	1/2	non	3	1	***10 *11 *1**
14	oui	oui	1	-	***1 *11**
15	non	non	0	-	***1**
16	non	non	0	-	***5**

***6** La surdétection provient des ambiguïtés qui demeurent encore sur la structure de ces phrases, en particulier sur le constituant *"un des bout de la chaîne de segments étiquetés"* dans la phrase 4 et sur le groupe verbal *"avons déjà défini des méthodes pendant la création des classes"*.

Cette dernière ambiguïté pourrait d'ailleurs se résoudre dans un certain nombre de cas avec une grammaire plus fine faisant intervenir les comportements rectionnels des verbes.

***7** Le nombre de reprises provient de ce que *"montre"* peut être un nom ou un verbe et génère de multiples hypothèses. Certaines solutions mettent en cause l'accord en personne entre le groupe nominal *"le système"* et le verbe *"fonctionne"*.

***8** L'erreur est décelée, mais le diagnostic est erroné puisque la faute est signalée sur le déterminant *"des"* alors que c'est le nom *"bout"* qui est mal accordé. Ceci provient du fait que les règles fixent a priori le constituant du groupe syntaxique sur lequel baser la correction. En particulier, la règle 506 (cf page 202) fixe le genre et le nombre du groupe nominal qu'elle construit sur celui du nom.

L'erreur effectuée sur la correction risque dans certains cas de se propager en provoquant d'autres erreurs non justifiées comme l'accord entre le groupe nominal et le verbe par exemple.

***9** Rien ne peut ressortir de l'analyse de la phrase à partir d'une grammaire qui n'accepte pas une suite de deux noms. Une reprise a mettre en place consisterait à intégrer a posteriori la règle qui accepte deux noms qui se suivent pour regarder si cela débloquerait le problème.

***10** Dans cette phrase, deux reprises portent sur la structure *"des langages"* comme groupe prépositionnel puis comme groupe nominal. La troisième reprise propose de construire la structure *"son dépendantes"* où *"son"* est considéré comme un nom.

***11** Les reprises sur erreur sont déclenchées tant que la structure de la phrase n'est pas déterminée.

5.5.3 Analyse immédiate

Nous mettons maintenant en œuvre une stratégie d'analyse immédiate. Nous introduisons dans nos points de comparaison le nombre de reprise sur fautes et le nombre de reprise sur analyse immédiate.

Type de stratégie	: **Immédiate, tolérante au niveau morphologique non tolérante au niveau syntaxique analyse syntaxique avec priorité aux règles** *Ou*
Base des règles	((800)(801)(802)(804 841 842)(815)(806)(807)(814)(809) (810)(811)(818)(819))
Confirm	= vrai
Prof règles morphologiques	= (¿ niveau 1)
Prof règles syntaxiques	= ()
Priorite_ou	= "ou"
ordre	= croiss

Les règles inverses sont rangées dans l'ordre : adjectifs, noms, verbes.

La règle de reprise 842 déclenche une reprise sur la recherche d'hypothèses sur les mots de la phrase.

(843	*reprise*	
(Si	*(objectif (anal_synt ?s enonce)))*	*si il y a échec sur l'analyse syntaxique de l'énoncé*
(Et_Si	*(distribue ?x mots_de_enonce))*	*et si ?x est un mot de l'énoncé*
(Alors	*(reprendre_objectif (définir_mot ?x))))*	*alors reprendre la recherche d'hypothèses sur ?x*

Phrase	*Détection des fautes*	*Surdétection*	*Reprises*	*Reprises sur faute*	*Reprises sur analyse immédiate*	*inutiles*	
1	oui	non	0	0	0	0	
2	oui	non	1	1	0	0	
3	oui	non	1	1	0	0	
4	oui	non	1	1	0	0	
5	oui	non	2	1	1	0	
6	oui	non	1	1	0	0	
7	oui	non	1	1	0	0	
8	oui	non	1	1	0	0	
9	oui	oui	1	1	0	0	
10	non	non	6	0	0	6	***12**
11	oui	oui	8	2	1	5	***13**
12	-	non	1	-	1	0	
13	1/2	non	6	1	0	5	***12 *1**
14	oui	oui	6	1	0	5	*** 12**
15	oui	non	8	0	0	8	***12**
16	non	non	0	0	0	0	***5**

***12** Les reprises sont déclenchées tant que la structure de la phrase n'est pas déterminée. Pour les phrases dont la structure ne peut pas être reconnue, soit parce que la grammaire n'est pas suffisamment couvrante, soit parce que la phrase comporte une erreur autre qu'une erreur d'accord ou de flexion, cela revient à redéclencher une reprise sur chacun des mots (autres que les mots outils). A terme, cela revient à une stratégie complète.

Un inconvénient majeur de ces règles et de nombreuses analyses effectuées dans des systèmes traitant le langage naturel est de se baser sur la reconnaissance de la structure de la phrase pour conclure au succès de l'analyse. Si on veut traiter des textes généraux, il paraît difficile de trouver une grammaire qui couvre leur totalité. Il faut donc définir des conditions d'arrêt de l'analyse ou modifier l'objectif global qui pour nous est **(nature prop enonce)**.

Dans le même ordre d'idée, le déclenchement d'une action de correction sur *"son"* dans la phrase 13 ne peut se faire que si on s'appuie sur la syntaxe et si on est sûr que c'est ce mot qui est à mettre en cause.

***13** Pour ce type de phrases, il serait intéressant de proposer des stratégies guidant la reprise vers les règles semblant les plus pertinentes vu la situation rencontrée.

Remarques

Nous avons testé cette même stratégie en changeant la stratégie d'analyse syntaxique. En particulier nous avons modifié l'ordre des règles inverses en nom, adj ,verbe et en verbe, adj, nom. Pour comparer les gains apportées par ces différentes stratégies, nous avons comparé le nombre de règles morphologiques filtrées, validées et appliquées dans chacun des cas. Sur les 16 phrases testées, l'ordre adj, nom, verbe est plus avantageux, mais le gain n'est pas vraiment significatif : on gagne en moyenne 3 règles morphologiques à filtrer sur une trentaine en moyenne par phrase. La stratégie à définir à ce niveau doit s'appuyer sur une étude préalable du type de textes à traiter et de la fréquence des différentes catégories morphologiques qui apparaissent.

Dans la stratégie suivante, nous apportons deux compléments. Le premier introduit le contrôle de cohérence avec les voisins grammaticaux avant de propager une hypothèse, le second porte sur la stratégie de reprise basée sur les constituants isolés.

5.5.4 Reprise sur constituants isolés

Reprise sur constituants isolés avant reprise générale

Type de stratégie : **Immédiate, tolérante au niveau morphologique**
non tolérante au niveau syntaxique
analyse syntaxique avec priorité aux règles *Ou*
contrôle de la compatibilité entre voisins
reprise sur constituants isol'es

Base des règles	((800)(801)(802)(804 844 845 841 842)(806)(807)(809)(810)(811) (818)(819)(814)(815)(816)(817))
Confirm	= vrai
Prof règles morphologiques	= (¿ niveau 1)
Prof règles syntaxiques	= ()
Priorite_ou	= "ou"
ordre	= croiss

Phrase	*Détection des fautes*	*Surdétection*	*Reprises*	*Reprises sur faute*	*Reprises sur analyse immédiate*	*inutiles*	
1	oui	non	0	0	0	0	
2	oui	non	1	1	0	0	
3	oui	non	1	1	0	0	
4	oui	non	1	1	0	0	
5	oui	non	2	1	1 "dernières"	0	
6	oui	non	1	1	0	0	***15**
7	oui	non	1	1	0	0	
8	oui	non	1	1	0	0	
9	oui	non	1	1	0	0	
10	non	non	6	0	0	6	***15**
11	oui	oui	6	2	1	3	***14**
12	-	non	1	-	1	0	***15**
13	1/2	non	6	1	0	5	
14	oui	oui	6	1	0	5	
15	oui	non	8	0	0	8	
16	non	non	0	0	0	0	

***14** On remarque ici le gain apporté par cette stratégie par la diminution du nombre de reprises amenant à la solution.

***15** Sur ces exemples, on veut montrer l'effet du test de compatibilité entre voisins. Ce test permet de ne pas créer de solutions superflues :

Dans la phrase 6, pas de propagation de l'hypothèse *"bonne"* comme nom.

Dans la phrase 12, la reprise sur *"demande"* est effectuée immédiatement avant de créer des solutions parasites avec *"demande"* comme nom.

Par contre dans la phrase 10, aucune hypothèse compatible avce ses voisins n'est de ce fait obtenue sur *"image"*.

Dans la stratégie suivante, nous essayons de limiter les reprises dans le cas de phrases aux structures erronées, en supprimant simplement les règles de reprise générale.

Reprise unique sur constituants isolés

La différence avec la stratégie précédente se définit au niveau de la règle 804 pour laquelle on a maintenant : (804 844 845).

Phrase	*Détection des fautes*	*Surdétection*	*Reprises*	*Reprises sur faute*	*Reprises sur analyse immédiate*	*inutiles*	
1	oui	non	0	0	0	0	
2	oui	non	1	1	0	0	
3	oui	non	1	1	0	0	
4	oui	non	1	1	0	0	
5	oui	non	2	1	1 "dernières"	0	
6	oui	non	1	1	0	0	
7	oui	non	1	1	0	0	
8	non	non	1	1	0	0	***16**
9	oui	oui	1	1	0	0	
10	non	non	1	0	0	1	
11	1/2	oui	3	1	0	2	
12	-	non	1	-	1	0	
13	1/2	non	2	1	0	1	
14	oui	oui	2	1	0	1	
15	oui	non	1	0	0	1	
16	non	non	0	0	0	0	

***16** La structure de la phrase n'est plus reconnue par cette stratégie parce que la reprise a été limitée aux seuls constituants isolés. Les premiers cycles d'inférences sur les règles syntaxiques ont construit les groupes *"frontières d'îles"* et *"représentée de la même manière"* et intègrent les constituants à mettre en cause pour l'échec.

Remarques

- Cette stratégie, si elle évite de nombreux traitements de reprise sur les phrases aux structures erronées, empêche la détection de fautes sur d'autres phrases correctement traitées par les stratégies précédentes.

  Le repérage des actions pertinentes doit donc être complété par d'autres critères.

- Nous avons voulu connaître si le fait d'accepter la structure nom-nom modifiait beaucoup les analyses précédentes. Pour cela nous avons appliqué l'avant-dernière stratégie en modifiant la grammaire pour qu'elle accepte la structure nom-nom.

  Sur la phrase 5, cette modification ajoute une ambiguïté et une reprise supplémentaire sur un accord possible entre *"ensemble"* et *"sémantiques"*.

  La phrase 10 permet ainsi d'être reconnue.

  Sur la phrase 6, la structure de la phrase est déterminée comme précédemment, mais le groupe *"bonne méthode"* est reconnu comme nom-nom.

  Sur la phrase 12, le résultat n'est plus obtenu qu'après 2 reprises sur em "change" puis *"demande"*.

5.5.5 Conclusion

En faisant l'hypothèse d'une grammaire plus développée, couvrant une part importante de la langue courante, et en particulier avec laquelle on puisse ajouter si nécessaire la règle sur la reconnaissance d'une suite de deux noms, la stratégie non tolérante au niveau syntaxique, avec reprise sur les constituants isolés, complétée par une reprise générale "aveugle" apporte des résultats qui nous paraissent les plus intéressants.

Ce n'est pas celle qui offre le moins de reprises possibles, en effet la stratégie testée à sa suite et qui ne fait intervenir qu'une reprise sur constituants isolés est meilleure de ce point de vue. Mais cette dernière présente par contre l'inconvénient de ne pas permettre de localiser des fautes suffisamment triviales pour faire douter l'utilisateur sur les compétences du système.

Il faut cependant moduler ce propos en se demandant si le fait d'augmenter la grammaire ne perturbe pas nos observations à propos des constituants isolés. Toute stratégie similaire, décrivant des critères de "anormalité" présenterait le même intérêt.

Conclusion

Nous avons choisi le domaine expérimental de la détection de fautes d'orthographe dans des textes écrits pour étudier l'intérêt et l'opportunité d'une analyse par affinements progressifs. Nous nous sommes situés dans un cadre général tel que l'utilisation de traitements de textes et pour cela :

- nous nous sommes abstenus de faire intervenir les connaissances propres au sujet traité par le texte (sémantique) ;
- nous avons systématisé le processus de détection en organisant notre système autour de deux grands niveaux de connaissances:
  - les connaissances sur la langue qui décrivent les règles à respecter ;
  - le savoir-faire qui permette de les utiliser pour l'analyse des énoncés et des fautes qu'ils contiennent.

Nous avons proposé un formalisme pour exprimer ce savoir-faire sous forme de règles de contrôle interprétables par un moteur d'inférence permettant le développement d'analyses par affinements progressifs.

Les expérimentations présentées, pour illustrer l'organisation de ces règles en règles de stratégie et règles de reprise et à compléter avec l'intégration d'un système de correction sur mots isolés, ont permis de montrer :

- la faisabilité et l'intérêt de développer des stratégies générales, notamment pour mettre en œuvre les connaissances les plus immédiates et les plus générales possibles ;
- l'avantage de prévoir des stratégies locales tenant compte de la spécificité du langage naturel et de l'objectif recherché ;
- l'intérêt pour cela de la structure progressive dégagée des règles du français pour adapter d'une part le traitement des énoncés erronés et en retirer des informations pertinentes , pour mettre en place d'autre part des stratégies d'analyse syntaxique ;

- les limites des analyses effectuées, en particulier les inconvénients d'une analyse dont le succès dépend de la reconnaissance de la structure de l'énoncé et la nécessité d'introduire des connaissances sémantiques pour traiter certains cas de figure ;
- les limites du formalisme de description des stratégies qui ne permet pas de modifier dynamiquement au cours des inférences le comportement d'une analyse et qui ne permet pas de traiter les erreurs non prévues.

Les difficultés majeures auxquelles vont se confronter les développeurs de logiciels de traitements de texte s'ils veulent élargir la détection des fautes à celle des fautes en contexte sont de deux catégories. D'une part, il y a le problème de description de la langue, en particulier celui de la conception d'une grammaire couvrant une part importante de la langue française (un consensus existe pour dire que cela parait impossible de couvrir entièrement la langue avec une grammaire de type hors-contexte) et celui de la part de sémantique à accorder à cette représentation. D'autre part, il faut envisager le problème du choix de stratégies ménageant efficacité et convivialité et sur lesquelles nous avons apporté quelques éléments. Il ne semble cependant pas déraisonnable que la détection des fautes sur les groupes nominaux et les phrases simples puisse être proposée à court terme. Mais cela correspond-il vraiment au besoin essentiel des rédacteurs que nous sommes ?

Bibliographie

[300 76] *300 trucs d'orthographe.* flash marabout, 1976.

[Antoniadis 84] G. Antoniadis. *Elaboration d'un système d'analyse morpho-syntaxique d'une langue naturelle. Application en informatique documentaire.* Thèse de 3ème cycle en mathématiques appliquées aux sciences sociales, Université des sciences sociales de Grenoble, Les cahiers du CRISS 5, 1984.

[Bescherelle] Bescherelle. *Le nouveau Bescherelle. 3 : La grammaire pour tous.* Hatier.

[Bescherelle 80] Bescherelle. *Le nouveau Bescherelle. 1 : L'art de conjuguer les verbes.* Hatier, 1980.

[Blair 60] C. R. Blair. A Program for Correcting Spelling Errors. *Information Control*, 3:60–67, 1960.

[Bobrow 77] D. G. Bobrow et T. Winograd. An Overview of KRL, a Knowledge Representation Language. *Cognitive Science*, 1(1):3–46, 1977.

[Castaing 88] J. Castaing, E. Bonte, P. Grandemange, S. Kolodziejckzyk, et F. Levy. Représentation des connaissances à profondeur variable. *Actes du 3ème colloque international Cognition et Connaissance*, pages 331–345, Association pour la Recherche Cognitive, Toulouse, mars 1988.

[Catach 82] Nina Catach. *L'orthographe.* Volume 685, Presses universitaires de France, que sais-je? édition, 1982.

[Catach 84a] N. Catach. *La phonétisation automatique du français.* Editions du CNRS, Paris, 1984.

[Catach 84b] Nina Catach. *Les listes orthographiques de base du français.* Nathan Recherche, 1984.

[Chailloux 86] J. Chailloux. *Le_Lisp de l'INRIA - Version 15.2.* I.N.R.I.A., Rocquencourt, 1986.

[Charpillet 85] F. Charpillet. Un système de reconnaissance de parole continue pour la saisie de textes lus. Thèse de l'université de Nancy 1, 1985.

[Chomsky 65] N. Chomsky. *Aspects Of Theory of Syntax.* MIT Press, Cambridge Mas., 1965. *Aspects de la théorie syntaxique*; traduction par J. C. Milner, Seuil, Paris, 1971.

[Chouraqui 85] E. Chouraqui, H. Farreny, D. Kayser, et H. Prade. Modélisation du raisonnement et de la connaissance. *Techniques et Sciences Informatiques*, 4(4):391–399, 1985.

[Cornew 68] R. W. Cornew. A Statistical Method of Spelling Correction. *Information and Control*, 12(2):79–93, 1968.

[Coulon 80] D. Coulon et D. Kayser. Un système de raisonnement à profondeur variable. *Actes du congrès AFCET informatique*, pages 517–527, Nancy, Novembre 1980.

[Coulon 82] D. Coulon et D. Kayser. La compréhension : un processus à profondeur variable. *Bulletin de psychologie*, XXXV(356):815–823, 1982.

[Damereau 62] F.J. Damereau. A Technique for Computer Detection and Correction of Spelling Errors. *Communication of the ACM*, 7(3):171–176, mars 1962.

[David 85] J. M. David. Analogie. *Actes du colloque Cognitiva 85*, Paris, 1985.

[Davidson 62] L. Davidson. Retrieval of Mispelled Names in an Airline's Passenger Record System. *Communication of the ACM*, 5(3):169–171, 1962.

[Davis 84] Randall Davis. Diagnostic Reasoning Based on Structure and Behavior. *Artificial Intelligence*, 24:347–410, 1984.

[Dubois 73] Jean Dubois et René Lagane. *La nouvelle grammaire du français.* Larousse, 1973.

[Durham 83] Ivor Durham, David Lamb, et James B. Saxe. Spelling Correction in User Interfaces. *Communication of the ACM*, 26(10):764–773, octobre 1983.

[Emond 85] J. C. Emond. Les types de règles et leur utilisation dans le domaine du diagnostic technique. *Actes des 5èmes journées internationales sur les systèmes experts et leurs applications*, pages 475–488, Avignon, 1985.

[Farreny 85] H. Farreny. Les systèmes experts. *Techniques Avancées de l'Informatique*, Cepadues Editions, Toulouse, 1985.

[Farreny 87] H. Farreny et M. Ghallab. Eléments d'intelligence artificielle. *Traité des nouvelles technologies. Série Intelligence artificielle*, Hermès, Paris, 1987.

[Fink 85] P. K. Fink. Control and Integration of Diverse Knowledge in a Diagnostic Expert System. *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, pages 426–431, Los Angeles, 1985.

[Fisher 76] E. G. Fisher. *The Use of Context in Character Recognition.* COINS T.R. no. 76-12, Department of Computer and Information Sciences, University of Massachussets, Amherst, juillet 1976.

[Fluhr 77] C. Fluhr. *Algorithmes à apprentissage et traitement automatique des langues.* Thèse de doctorat es sciences, Université de Paris-sud, 1977.

[Fouqueré 86] C. Fouqueré. Analyse tolérante de textes en langage naturel. *Actes du séminaire Gréco-Galf de la communication parlé du CNRS : "Lexique et traitement automatique des langages"*, pages 67–74, Toulouse, 1986.

[Fouqueré 87] C. Fouqueré. Système de correction de textes orienté acteur. *Actes du 6ème congrès AFCET Reconnaissance des Formes et Intelligence Artificielle*, pages 899–912, Antibes, 1987.

[Fouqueré 88] C. Fouqueré. *Système d'analyse tolérante du langage naturel.* Thèse de l'Université de Paris-Nord, 1988.

[Fournier 84] J.P. Fournier. *"Sauvetage" de raisonnements en langage naturel.* Thèse de 3ème cycle, Université de Paris-Sud, 1984.

[Fox 81] M. S. Fox. Reasoning with Incomplete Knowledge in a Ressource-Limited Environment : Integrating Reasoning and Knowledge Acquisition. *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 313–318, Vancouver, 1981.

[Granger 84] V. Granger. Correction de fautes de grammaire et génération de langage naturel en PROLOG. M. Dincbas, éditeur, *Actes du séminaire de Programmation en Logique*, pages 139–149, CNET - Lannion, Plestin-les-Greves, 1984.

[Grevisse 75] M. Grevisse. *Le bon usage.* Duculot, 10ème édition, 1975.

[Gruaz 86] C. Gruaz. Forme et fonction d'un composant morphémique en intelligence artificielle. *Actes du séminaire Gréco-Galf de la communication parlée du CNRS: "Lexique et traitement automatique des langages"*, pages 75–85, Toulouse, janvier 1986.

[Hall 80] A. V. Hall et G. R. Dowling. Approximate String Matching. *Computing surveys*, 12(4):381–402, décembre 1980.

[Hayes 80] P. J. Hayes et G. V. Mouradian. Flexible Parsing. *18th Annual Meeting of the ACL*, pages 97–103, Pennsylvanie, U.S.A., june 1980.

[Hayes 84] P. J. Hayes. Entity-oriented parsing. *Proceedings of the COLING'84*, pages 212–217, Stanford University, U.S.A., July 1984.

[Heidorn 82] G.E. Heidorn, K. Jensen, L.A. Miller, R.J. Byrd, et M.S. Chodorow. The EPISTLE Text-Critiquing System. *IBM system journal*, 21(3), 1982.

[Karoubi 86] M. Karoubi. *Système de raisonnement en langage naturel guidé par un raisonnement caricatural.* Thèse de 3ème cycle, Université de Paris-Sud, Centre d'Orsay, 1986.

[Kayser 88] D. Kayser. Le raisonnement à profondeur variable. Teknea, éditeur, *Actes des journées nationales du PRC-GRECO Intelligence Artificielle*, pages 109–136, Toulouse, 1988.

[LallichBoidin 86] G. Lallich-Boidin. *Analyse syntaxique automatique du français : Applications à l'indexation automatique.* Thèse de doctorat d'état, Université des Sciences Sociales de Grenoble, 1986.

[Larousse 82] Larousse. *Larousse de l'orthographe.* Larousse, 1982.

[Lauriere 82] J. L. Laurière. Représentation et utilisation des connaissances. *Techniques et Sciences Informatiques*, 1(1,2), 1982.

[Lenhert 79] W. Lenhert et Y. Wilks. A Critical Perspective on KRL. *Cognitive Science*, 3(1):1–28, 1979.

[Lenvenshtein 66] V. I. Lenvenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Sov. Phys.-Dokl*, 10(8):707–710, 1966.

[Marcus 80] M. P. Marcus. *A Theory of Syntactic Recognition for Natural Language.* M.I.T. Press, Cambridge, Mass, 1980.

[Minton 85] S. Minton, P. J. Hayes, et J. Fain. Controlling Search in Flexible Parsing. *Proceedings of the 9th IJCAI*, pages 785–787, Los Angeles, U.S.A., August 1985.

[Morgan 70] H.L. Morgan. Spelling Correction in Systems Programs. *Communications of the ACM*, 13(2):90–94, février 1970.

[Morris 75] R. Morris et L. L. Cherry. Computer Detection of Typographical Errors. *IEE Transactions on Professional Communications*, PC-18(1):54–64, 1975.

[Muth 77] F.E. Muth et A.L. Tharp. Correcting Human Error in Alphanumeric Terminal Input. *Information Processing and Management*, 13:329–337, 1977.

[Pavard 85] B. Pavard. La conception des systèmes de traitement de textes. *Intellectica*, 1(1):37–68, 1985.

[Pereira 80] F. C. Pereira et D. H. D. Warren. Definite Clause Grammars for Language Analysis. A Survey of the Formalism and a Compàrison with Augmented Transition Networks. *Artificial Intelligence*, 13:231–278, 1980.

[Pérennou 86] Guy Pérennou, Pierre Daubeze, et François Lahens. La vérification et la correction automatique. *T.S.I.*, 5(4):285–305, 1986.

[Peterson 80] J. L. Peterson. Computer Programs for Spelling Detection : An Experiment in Program Design. *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 1980.

[Pierrel 81] J. M. Pierrel. *Etude et mise en œuvre de contraintes linguistiques en compréhension automatique du discours continu.* Thèse d'état, Université de Nancy 1, 1981.

[Pitrat 84] J. Pitrat. Connaissances et métaconnaissances déclaratives. *Actes du colloque ARC*, pages 439–450, 1984.

[Prade 83] H. Prade. A Synthetic vieuw of Approximate Reasoning Techniques. *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pages 130–136, Karlsruhe, West Germany, August 1983.

[Richard 86] Danielle Richard et Guy Lapalme. Un système de correction automatique des accords des participes passés. *TSI*, 5(4):307–319, 1986.

[Riseman 74] E. M. Riseman et A. R. Hanson. A Contextual Postprocessing System for Error Correction Using binary n-grams. *IEEE Trans. Comput.*, C-23(5):480–493, 1974.

[Sabah 88] G. Sabah. *L'intelligence artificielle et le langage : représentation des connaissances.* Hermès, 1988.

[Schank 79] R. C. Schank. Interestingness : Controlling Inferences. *Techniques et Sciences Informatiques*, 12(3):273–297, 1979.

[Sedogbo 83] C. Sedogbo. *Fonctions syntaxiques dans un système question-réponse utilisant des règles contextuelles.* Thèse de 3ème cycle, Université de Paris-Sud, 1983.

[Sislenchi 85] P. Sislenchi et D. Vernet. Extase : Exemple de traitement d'alarmes par système expert. *Actes du 5ème congrès AFCET Reconnaissance des Formes et Intelligence Artificielle*, pages 943–952, Grenoble, 1985.

[Thimonnier 79] Renè Thimonnier et Jean Desmeuzes. Les 30 problèmes de l'orthographe en 76 leçons. *Marabout Service*, marabout, 1979.

[VandeVelde 85] W. Van de Velde. Naïve causal reasoning for diagnosis. *Actes des 5èmes journées internationales sur les systèmes experts et leurs applications*, Avignon, 1985.

[Veronis 88] J. Véronis et J. P. Fournier. Traitement des erreurs dans la communication homme-machine en langage naturel. *Actes des journées nationales du GRECO-PRC Communication homme-machine*, pages 129–149, Paris, 1988.

[Viterbi 67] A. J. Viterbi. Errors Bounds on Conventional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Trans. Inf. Theory.*, IT(13):260–269. April 1967.

[Weischedel 83] R. Weischedel et N. Sondheimer. Meta-Rules as a Basis for Processing Ill-Formed Input. *American journal of Computational Linguistics*, 9(3-4):161–177, 1983.

[Woods 70] W. A. Woods. Transitions Network Grammar for Natural Language. *Journal of the ACM*, 13(10):591–606, 1970.

Annexe 1 : Exemples de phrases erronées extraites du corpus de fautes

Les phrases erronées recensées dans notre corpus sont d'origines diverses. Elles ont été extraites :

- d'articles de quotidiens régionaux et nationaux ;
- d'articles rédigés par des chercheurs du laboratoire, dont une personne étrangère ;
- de dictées et rédactions d'élèves de CPA (classes de préparation à l'apprentissage) ;
- de divers documents comme des magazines, des énoncés d'examens etc...

Nous en donnons ci-dessous un échantillon présentant principalement les erreurs auxquelles nous nous sommes intéressés.

1. "L'environnement de conception génére(ère) un certain nombre de directives"
2. "Cette excellentes() composition enchante les productions de jeux télévisés"
3. "Le graphe est un ensemble de segments de droite obtenus par approximation polygonales() des contours"
4. "On prend un segment connecté à un des bout_(s) de la chaîne de segments étiquetés"
5. "Ces dernières constitue(ent) l'ensemble des équations sémantiques"
6. "La bonne méthode est recherché_(e) en fonction du seul type du receveur"
7. "Nous avons déjà définis() des méthodes pendant la création des classes"
8. "Le modèle contient différentes frontières d'îles représentés(es) de la même manière"

9. "Les sections qui suivent décrivent de manière détaillée le_(s) fonctionnalités offertes par le système aux utilisateurs des environnements générés"
10. "Le nombre d'indices image_(s?) est généralement important"
11. "Les résultats expérimentaux montre(ent) que le système fonctionne bien dans des situation_(s) de recouvrement partiel"
12. "La sociétés() des agents de change demande des sanctions"
13. "Les descriptions des modules son_(t) dépendantes de la syntaxe des langage_(s)"
14. "Le principe des fonctions génériques est extrè(ê)mement sympathique"
15. "Ces contraintes que nous _(ne) supposerons que binaires créent donc un réseau d'arcs sur ces points"
16. "Cet article portait sur le comportement d'une variable hérité_(e)"
17. "Un tel choix est étroitement lié à l'applicaiton(tion)"
18. "Un réseau local nécessite d'important_(s) moyens techniques humains pour son installation"
19. "Les grandes ventes de tableaux connu_(s) se déroulent à Londres"
20. "Une discussion de cette technique est faî(i)te dans cet article"
21. "L'ensemble d'équations sémantiques décrit l'interprétation que les différents outil_(s) de l'environnement donnent au langage"
22. "Une classes hérite des variables de ses métaclasses"
23. "Un segment parasite est inclu_(s) dans un groupe de segments parallèles"
24. "Notre but est la mise à jour incrémentale d'un modèle tridimensionnel constitués() de segments de droites"
25. "Ce jeune chef d'entreprise rémois part à la conquête(ête) de l'Amérique"
26. "Il a l'impression de travailler sur un éditeur de textes classiques()"
27. "Ce sont de fausse_(s) pierres de polystyrène collées sur une armature"
28. "Cette méthode permet d'éliminer suffisam(mm)ent d'hypothèses"
29. "Les opérations réalisées par les deux charges sont de nature_(s) différentes"

30. "L'un des bateaux était tiré par une() chalutier de la marine royale"

31. "Cette ferme auberge risque de compromettre le(la) situation du jeune restaurateur du village"

32. "Le budget prévu pour cette année s'élé(è)ve à des millions de dollars"

33. "Il estime donc normal la commission perçue pour cette opérations()"

34. "La compagnie financière donne l'ordre à la banque de vendre tout(s) les titres"

35. "Le livre s'adresse à tous ceux qui veule(ent) connaître les étoiles, les comètes , les constellations"

36. "Un ciel couvert de cirrus laisse présager des vents fort_(s)"

37. "Le problème de reconnaissance d'objets se ramène à une mise en correspondance entre une certaine représentation de l'image et une représentation des objets connu_(s) par le système"

38. "Les instances, les classe_(s) et les métaclasses s'instancient de la même manière"

39. "Ceci restreint de façon notoire la combinatoire après les premier_(s) choix"

40. "Il a été ministre des affaires étrangère_(s) de la république d'Irlande"

41. "Des timbres à la feuille, dessiné_(s) et gravé_(s) en taille douce par J. G."

42. "Dans le ciel de velours violet, les étoiles brillait(aient), innombrables"

43. "Je descendait(ais) allumer le grand feu de bois"

44. "Nous avons employée() une méthode assez similaire"

45. "La qualité de la meilleure transformation est supérieur_(e) à un certain seuil"

46. "Des algorithmes efficaces utilisant la consistence d'arcs ont été élaborée()s pour ce dernier problème"

47. "L'interprétation des formes complexes revient dans la majorité des cas à affecter une étiquette des groupes d'indices dans l'images()"

48. "Un sous-graphe complet est appelée() une clique d'où le nom donné à cet(ette) méthode"

49. "Cette condition est très forte et gé(ê)nante"

50. "G. à(a) réalisé une implantation de C. en O."

51. "Il est ainsi possible de détecter certaine_(s) anomalies dans la liste des arguments lors de l'appel d'une fonction générique"

52. "On peut reprocher a(à) la norme actuelle une trop grande complexité pour être utilisée par un programmeur quelconque"

53. "La valeur de cet attribut repre(é)sente l'ensemble des identificateurs utilisés dans les instructions d'un programme"

54. "La méthode est souvent cité_(e) comme exemple représentatif de système utilisant les cliques maximales"

55. "Les donneé(ée)s fournies par les capteurs des systèmes de vision ont la caractéristique banale d'êtres() difficiles à interpréter"

56. "Aucune poursuite engagée contre des personnes considérés(es) dangereuses"

57. "On peut aussi recherche(er) par exemple des droites dont la paramétrisation est des plus simples"

58. "Chaque objet à(a) le contrôle de ses variables"

59. "Les différentes possibilités de correspondance entre les indices image_(s) et les formes primitives du modèle peuvent être représentés(es) par un arbre formé par les hypothèses de correspondance"

60. "Le_(s) métaclasses Smalltalk-80 permettent elles aussi de définir des variables d'instance des métaclasses"

61. "Je travail(aille) mieux quand j'ai écouté le chant des oiseaux"

62. "Ces connaissances sont des relations entre formes et entre règles qui ont été introduite_(s) par M. pour l'analyse de formes structurées"

63. "Les propriétés des relations pour décrire la(le) domaine et le problème"

Annexe 2 : Exemples de règles morphologiques

5.6 Introduction

5.6.1 Références

Nous fournisssons ci-dessous un ensemble de règles morphologiques utilisées en particulier pour l'analyse des exemples présentés dans les chapitres précédents.

Pour effectuer nos expérimentations, nous avons extrait une *expertise* de manuels traitant, comme leurs titres l'indiquent, du *Bon usage*, de l'*orthographe* en général, des *conjugaisons*, de la *grammaire* : *[Dubois 73], [Grevisse 75], [Bescherelle 80], [Larousse 82], [Bescherelle].*

Selon l'objectif visé, pédagogie, synthèse, exhaustivité, la précision des lois exposées varie d'un ouvrage à l'autre. Nous avons dû en faire une synthèse. Le *[Larousse 82]*, plus détaillé que *[Bescherelle]* mais moins exhaustif que *[Grevisse 75]* nous a souvent servi de référence pour les connaissances sur la morphologie.

Il est bien évident que ces choix, subjectifs, pourront être discutés. Toute autre conception de la langue qui la décrirait de façon progressive (règles générales simples, règles particulières de plus en plus restrictibles) devrait convenir. Le formalisme LISP pour écrire ces règles n'est pas un handicap dans la mesure où nous nous sommes limités à quelques structures de programmes et qu'il serait possible alors de réaliser un compilateur de règles, ces dernières étant rédigées dans un format plus convivial pour le concepteur.

5.6.2 Elaboration des règles

Propagation des attributs

Nous avons systématiquement ajouter dans toute règle morphologique la propagation des attributs grammaticaux d'un radical vers ses formes fléchies : genre, nombre, personne, classe ... qui n'est pas explicitement décrite dans les manuels.

Fusionner la description des mêmes phénomènes

- Certaines règles décrivant le même comportement flexionnel ont été fusionnées. Par exemple les règles qui décrivent la construction du féminin des noms en "n", "t", "d" et "s" sont

regroupées en une seule règle (règle 26). La fonction **chaque** utilisée dans ce cas permet d'instancier la règle pour chacune des désinences d'une liste donnée. Le rôle de la fonction **les** est d'extraire les informations associées à cette désinence.

Soit une liste de la forme :

liste = $((des_1(prop_1 val_{11})(prop_2 val_{12})...)(des_2(prop_1 val_{21})(prop_2 val_{22})...)...)$

`(chaque ?d liste)` donne successivement à `?d` la valeur des_1, des_2...

`(les liste` $prop_i$ des_j`)` rend la valeur val_{ji}

Par exemple la règle 26 ci-dessous revient aux 4 règles suivantes :

```
(26 init
      (SI   (unif mot (conc ?x (chaque ?d L_TER_FEM1))))
      (ET_SI (nature nom mot)
             (classe anime mot)
             (genre masc mot)
             (nombre sing mot))
      (ALORS (egal ?u (conc ?x ?d "e"))
             (ajouter_hyp ((nature nom)(nombre sing)(genre fem))
                          ?u))
      (SAUF (les L_TER_FEM1 exceptions_fem ?d)))
```

Avec :

```
L_TER_FEM1 = '(("n" (exceptions_fem 6 29 16 18))
               ( "t" (exceptions_fem 7 8))
               ("s" (exceptions_fem 10 20 19))
               ("d")))
```

```
(26(1) init
      (SI   (unif mot (conc ?x "n")))
      (ET_SI (nature nom mot)
             (classe anime)
             (genre masc mot)
             (nombre sing mot))
      (ALORS (egal ?y (conc ?x "ne"))
             (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                          ?y))
      (SAUF (6 29 16 18)))

(26(2) init
      (SI   (unif mot (conc ?x "t")))
      (ET_SI (nature nom mot)
             (classe anime mot)
             (genre masc mot)
             (nombre sing mot))
      (ALORS (egal ?y (conc ?x "te"))
             (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                          ?y))
      (SAUF (7 8)))
```

```
(26(3) init
      (SI   (unif mot (conc ?x "s")))
      (ET_SI (nature nom mot)
             (classe anime mot)
             (genre masc mot)
             (nombre sing mot))
      (ALORS (egal ?y (conc ?x "se"))
             (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                          ?y))
      (SAUF (10 20 19)))

(26(4) init
      (SI   (unif mot (conc ?x "d")))
      (ET_SI (nature nom mot)
             (classe anime mot)
             (genre masc mot)
             (nombre sing mot))
      (ALORS (egal ?y (conc ?x "de"))
             (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                          ?y))
      (SAUF))
```

- Une règle de conjugaison regroupe 3 formes conjuguées correspondant soit aux formes du singulier soit aux formes du pluriel.

Lorsque la clause *Sauf* ou la clause *Ou* est omise elle est vide.

5.7 Exemples de Règles sur la construction du pluriel

Exemples de Règle sur la construction du pluriel des noms

```
(50 init_morpho
      (SI    (unif mot ?x))
      (ET_SI (nature nom mot)
             (nombre sing mot)
             (genre ?g mot))
      (ALORS (egal ?y (conc ?x "s"))
             (ajouter_hyp ((nature nom)(genre ?g))(nombre plur))
                          ?y))
      (SAUF (51 56 57 59 ...))
      (OU   (58)))

(51 init_morpho
      (SI   (unif mot (conc ?x "au")))
      (ET_SI (nature nom mot)
             (nombre sing mot)
             (genre ?g mot))
      (ALORS (egal ?y (conc ?x "aux"))
             (ajouter_hyp ((nature nom)(nombre plur)(genre ?g))
                          ?y))
      (SAUF (60)))
```

```
(60 init_morpho
      (SI   (appartient L_TER_AUS mot)
            (unif mot (conc ?x "au")))
      (ET_SI)
      (ALORS (egal ?y (conc ?x "aus"))
             (ajouter_hyp ((nature nom)(nombre plur)(genre masc))
                          ?y)))

(56 init_morpho
      (SI   (unif mot (conc ?x "eu")))
      (ET_SI (nature nom mot)
             (nombre sing mot)
             (genre ?g mot))
      (ALORS (egal ?y (conc ?x "eux"))
             (ajouter_hyp ((nature nom)(nombre plur)(genre ?g))
                          ?y))
      (SAUF (62))
      (OU   (61)))

(62 init_morpho
      (SI   (appartient L_TER_EUS mot)
    (unif mot (conc ?x "eu")))
      (ET_SI)
      (ALORS (egal ?y (conc ?x "eus"))
             (ajouter_hyp ((nature nom)(nombre plur)(genre masc))
                          ?y)))

(61 init_morpho
      (SI    (unif mot "lieu"))
      (ET_SI)
      (ALORS (ajouter_hyp ((nature nom)(nombre plur)(genre masc))
                          "lieus")))

(57 init_morpho
      (SI    (unif mot (conc ?x (chaque ?d L_TER_PLUR_INV))))
      (ET_SI (nature nom mot)
             (nombre masc mot)
             (genre ?g mot))
      (ALORS  (ajouter_hyp ((nature nom)(nombre plur)(genre ?g))
                           mot)))

(59 init_morpho
    (SI    (unif mot (conc ?x "al")))
    (ET_SI (nature nom mot)
           (nombre sing mot)
           (genre ?g mot))
    (ALORS (egal ?y (conc ?x "aux"))
           (ajouter_hyp ((nature nom)(genre ?g)(nombre plur))
                        ?y))
    (SAUF (54))
    (OU   (55)))
```

```
(54 init_morpho
      (SI    (appartient L_PLUR_AL mot)
             (unif mot (conc ?x "al")))
      (ET_SI)
      (ALORS (egal ?y (conc ?x "als"))
             (ajouter_hyp ((nature nom)(nombre plur)(genre masc))
                          ?y)))

(55 init_morpho
      (SI    (appartient L_PLUR_AL_2F mot)
             (unif mot (conc ?x "al")))
      (ET_SI)
      (ALORS (egal ?y (conc ?x "als"))
             (ajouter_hyp ((nature nom)(nombre plur)(genre masc))
                          ?y)))
```

L_PLUR_AL = '("aval" "bal" "cal" "cantal" "carnaval" "cérémonial" "chacal" "choral" "festival" "gavial" "gayal" "narval" "nopal" "pal" "récital" "régal" "rorqual" "santal" "sisal" "tincal" "trial"))

L_TER_PLUR_INV = '("s" "x" "z"))

L_PLUR_AL_2F = '("étal" "final" "idéal" "val"))

L_TER_AUS = '("landau" "sarrau"))

L_TER_EUS = '("émeu" "bleu"))

Exemples de règles sur la construction du pluriel des adjectifs

```
(70 init_morpho
      (SI    (unif mot ?x))
      (ET_SI (nature adj mot)
             (nombre sing mot)
             (genre ?g mot))
      (ALORS (egal ?y (conc ?x "s"))
             (ajouter_hyp ((nature adj)(genre ?g)(nombre plur))
                          ?y))
      (SAUF (71 76 73 ...)))

(71 init_morpho
      (SI   (unif mot (conc ?x "au")))
      (ET_SI (nature adj mot)
             (nombre sing mot)
             (genre ?g mot))
      (ALORS (egal ?y (conc ?x "aux"))
             (ajouter_hyp ((nature adj)(nombre plur)(genre ?g))
                          ?y)))
```

```
(76 init_morpho
     (SI   (unif mot (conc ?x "eu")))
     (ET_SI (nature adj mot)
            (nombre sing mot)
            (genre ?g mot))
     (ALORS (egal ?y (conc ?x "eux"))
            (ajouter_hyp ((nature adj)(nombre plur)(genre ?g))
                 ?y))
     (SAUF (72)))

(72 init_morpho
     (SI   (appartient L_EUS_ADJ mot)
           (unif mot (conc ?x "eu")))
     (ET_SI)
     (ALORS (egal ?y (conc ?x "eus"))
            (ajouter_hyp ((nature adj)(nombre plur)(genre masc))
                         ?y)))

(73 init_morpho
   (SI    (unif mot (conc ?x "al")))
   (ET_SI (nature adj mot)
          (nombre sing mot)
          (genre ?g mot))
   (ALORS (egal ?y (conc ?x "aux"))
          (ajouter_hyp ((nature adj)(genre ?g)(nombre plur))
                       ?y))
   (SAUF (74))
   (OU   (75)))

(74 init_morpho
     (SI    (appartient L_PLUR_AL_ADJ mot))
     (ET_SI (nature adj mot))
     (ALORS (egal ?y (conc mot "s"))
            (ajouter_hyp ((nature adj)(nombre plur)(genre masc))
                         ?y)))

(75 init_morpho
     (SI    (appartient L_PLUR_AL_2F_ADJ mot)
            (unif mot (conc ?x "al")))
     (ET_SI)
     (ALORS (egal ?y (conc ?x "als"))
            (ajouter_hyp ((nature adj)(nombre plur)(genre masc))
                         ?y)))
```

```
L_PLUR_AL_ADJ     =  '("bancal" "fatal" "natal" "naval" "nymphal" "tonal"))
L_PLUR_AL_2F_ADJ  =  '( "austral" "banal" "boréal" "causal" "final" "glacial"
                        "idéal" "jovial" "pascal"))
L_EUS_ADJ         =  '("bleu" "feu"))
```

Règles sur le pluriel des participes passés

```
(80 init_morpho
      (SI    (unif mot ?x))
      (ET_SI (nature pp ?x)
             (nombre sing ?x)
             (genre ?g ?x))
      (ALORS (egal ?y (conc mot "s"))
             (ajouter_hyp ((nature pp)(genre ?g)(nombre plur))
                          ?y))
      (SAUF (81)))

(81 init_morpho
      (SI    (unif mot (conc ?x "s")))
      (ET_SI (nature pp mot)
             (nombre masc mot)
             (genre ?g mot))
      (ALORS (ajouter_hyp ((nature pp)(nombre plur)(genre ?g))
                          mot)))
```

5.8 Exemples de règles sur la construction du féminin

Exemples de règles sur la construction du féminin des noms

La classe anime caractérise les noms qui accepte un variante au féminin.

```
(1 init_morpho
          (SI    (unif mot (conc ?y (chaque ?d VOYELLE))))
          (ET_SI (nature nom mot)
 (classe anime mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc mot "e"))
                 (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              ?y))
          (SAUF (les VOYELLE  exceptions_fem ?d )))

(5 init_morpho
          (SI   (unif mot (conc ?x "ou")))
          (ET_SI (nature nom mot)
 (classe anime mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x "olle"))
                 (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              ?y))
          (SAUF (13)))
```

```
(2 init_morpho
          (SI    (unif mot (conc ?x (chaque ?d L_TER_FEM2))))
          (ET_SI (nature nom mot)
                 (classe anime mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x (les L_TER_FEM2 fem ?d)))
                 (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              ?y))
          (SAUF (les L_TER_FEM2 exceptions_fem ?d ))
          (OU   (les L_TER_FEM2 regles_ou ?d)))

(9 init_morpho
          (SI   (unif mot (conc ?x "al")))
          (ET_SI (nature nom mot)
                 (classe anime mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x "ale"))
                 (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                               ?y)))

(11 init_morpho
          (SI   (appartient L_X mot))
          (ET_SI)
          (ALORS (egal ?y (les L_X fem  mot))
          (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              ?y)))

(12 init_morpho
          (SI   (appartient L_C mot))
          (ET_SI)
          (ALORS (egal ?y (les L_C fem mot))
          (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              ?y)))

(26 init_morpho
         (SI    (unif mot (conc ?x (chaque ?d L_TER_FEM1))))
         (ET_SI (nature nom mot)
                (classe anime mot)
                (genre masc mot)
                (nombre sing mot))
         (ALORS (egal ?y (conc ?x ?d "e"))
                (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                             ?y))
         (SAUF (les L_TER_FEM1 exceptions_fem ?d)))
```

```
(6 init_morpho
          (SI    (unif mot (conc ?x "on")))
          (ET_SI (nature nom mot)
                 (classe anime mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x "onne"))
                 (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              ?y))
          (SAUF (15 17))
          (OU (28)))

(15 init_morpho
          (SI    (unif mot "mormon"))
          (ET_SI)
          (ALORS (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              "mormone")))

(17 init_morpho
          (SI   (unif mot "compagnon"))
          (ET_SI)
          (ALORS (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                               "compagne")))

(28 init_morpho
          (SI   (appartient L_ON2 mot)
        (unif mot (conc ?x "on")))
          (ET_SI)
          (ALORS (egal ?y (conc ?x  "one"))
                 (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              ?y)))
(7 init_morpho
          (SI   (unif mot (conc ?x "et")))
          (ET_SI (nature nom mot)
                 (classe anime mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x "ette"))
          (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              ?y))
         (SAUF (31)))

(31 init_morpho
          (SI   (unif mot "prèfet"))
          (ET_SI)
          (ALORS (ajouter_hyp ((nature nom)(genre fem)(nombre sing))
                              "préfète")))
```

```
L_TER_FEM1  =  '(("n" (exceptions_fem 6 29 16 18))
                ( "t" (exceptions_fem 7 8))
                ("s" (exceptions_fem 10 20 19))
                ("d")))
L_TER_FEM2  =  '(("x" (fem "se")(exceptions_fem 11))
                ( "f" (fem "ve"))("p" (fem "ve"))
                ("er" (fem "ère"))
                ("eur" (fem "euse")(exceptions_fem 21 22 24)(regles_ou 25))
                ("l" (fem "lle")(exceptions_fem 9))
                ("c" (fe "que")(exceptions_fem 12))
                ("g" (fem "gue"))))
L_ON2       =  '("lapon" "letton" "nippon"))
L_C         =  '(("duc" (fem "duchesse")) ("grec" (fem "grecque"))))
```

Exemples de règles sur la construction du pluriel des adjectifs

```
(101 init_morpho
        (SI    (unif mot (conc ?y (chaque ?d VOYELLE_ADJ))))
        (ET_SI (nature adj mot)
               (genre masc mot)
               (nombre sing mot))
        (ALORS (egal ?y (conc mot "e"))
               (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                            ?y))
        (SAUF (les VOYELLE_ADJ  exceptions_fem ?d )))

(105 init_morpho
        (SI   (unif mot (conc ?x "ou")))
        (ET_SI (nature adj mot)
               (genre masc mot)
               (nombre sing mot))
        (ALORS (egal ?y (conc ?x "olle"))
               (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                            ?y))
        (SAUF (113)))

(113 init_morpho
        (SI   (appartient L_OU mot)
              (unif mot (conc ?x "ou")))
        (ET_SI)
        (ALORS (egal ?y (conc ?x "oue"))
               (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                            ?y)))
```

```
(102 init_morpho
          (SI    (unif mot (conc ?x (chaque ?d L_TER_FEM2_ADJ))))
          (ET_SI (nature adj mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x (les L_TER_FEM2_ADJ fem ?d)))
                 (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                              ?y))
          (SAUF (les L_TER_FEM2_ADJ exceptions_fem ?d ))
          (OU   (les L_TER_FEM2_ADJ regles_ou ?d)))

(109 init_morpho
          (SI   (unif mot (conc ?x "al")))
          (ET_SI (nature adj mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x "ale"))
                 (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                              ?y)))

(111 init_morpho
          (SI   (appartient L_X_ADJ mot))
          (ET_SI)
          (ALORS (egal ?y (les L_X_ADJ fem mot))
                 (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                              ?y)))

(112 init_morpho
          (SI   (appartient L_C_ADJ mot))
          (ET_SI)
          (ALORS (egal ?y (les L_C_ADJ fem mot))
                 (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                              ?y)))

(126 init_morpho
         (SI    (unif mot (conc ?x (chaque ?d L_TER_FEM1_ADJ))))
         (ET_SI (nature adj mot)
                (genre masc mot)
                (nombre sing mot))
         (ALORS (egal ?y (conc ?x ?d "e"))
                (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                             ?y))
         (SAUF (les L_TER_FEM1_ADJ exceptions_fem ?d)))

(106 init_morpho
          (SI    (unif mot (conc ?x "on")))
          (ET_SI (nature adj mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x "onne"))
                 (ajouter_hyp ((nature adj)(genre fem)(nombre sing)) ?y))
          (SAUF (115))(OU (128)))
```

```
(128 init_morpho
          (SI   (appartient L_ON2 mot)
                (unif mot (conc ?x "on")))
          (ET_SI)
          (ALORS (egal ?y (conc ?x "one"))
                 (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                              ?y)))

(115 init_morpho
          (SI   (unif mot "mormon"))
          (ET_SI)
          (ALORS (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                              "mormone")))

(107 init_morpho
          (SI   (unif mot (conc ?x "et")))
          (ET_SI (nature adj mot)
                 (genre masc mot)
                 (nombre sing mot))
          (ALORS (egal ?y (conc ?x "ette"))
                 (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                              ?y))
          (SAUF (118)))

(118 init_morpho
          (SI   (appartient L_ET mot)
                (unif mot (conc ?x "et")))
          (ET_SI)
          (ALORS (egal ?y (conc ?x "` ete"))
                 (ajouter_hyp ((nature adj)(genre fem)(nombre sing))
                              ?y)))
```

```
VOYELLE_ADJ     =  '(("a" (exceptions_fem 114))
                    ("é")
                    ("i" (exceptions_fem 114))
                    ("u" (exceptions_fem 104 105 114))
                    ("o" (exceptions_fem 114))
                    ("y")))
L_TER_FEM1_ADJ  =  '(("n" (exceptions_fem 106 129 116))
                    ( "t" (exceptions_fem 107 108))
                    ("s" (exceptions_fem 110 117 119 122))
                    ("d")))
L_TER_FEM2_ADJ  =  '(("x" (fem "se")(exceptions_fem 111))
                    ( "f" (fem "ve"))("p" (fem "ve"))
                    ("er" (fem "ère"))
                    ("eur" (fem "euse")(exceptions_fem 123 124)(regles_ou 125))
                    ("l" (fem "lle")(exceptions_fem 109))
                    ("c" (fe "que")(exceptions_fem 112 103))
                    ("g" (fem "gue"))))
L_X_ADJ   =  '(("faux" (fem "fausse"))("roux" (fem "rousse")) ("vieux" (fem "vieille"))))
L_ET      =  '("complet" "concret" "désuet" "discret" "incomplet" "indicret" "inquiet" "re-
             plet" "secret"))
L_ON2     =  '("lapon" "letton" "nippon"))
L_C_ADJ   =  '(("sec" (fem "sèche"))("franc" (fem "franche")) ("blanc" (fem "blanche"))))
```

5.9 Exemples de règles sur la conjugaison des verbes

```
(151 init_morpho
       (SI    (unif mot (conc ?x "er")))          ; "chanter"
       (ET_SI (nature infinitif mot)
              (groupe 1 mot)
              (classe ?c mot))
       (ALORS (egal ?y (conc ?x "e"))
              (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                            (personne 1)(classe ?c))
                            ?y)
              (egal ?z (conc ?x "es"))
              (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                            (personne 2)(classe ?c))
                           ?z)
              (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                            (personne 3)(classe ?c))
                           ?y))
       (SAUF (154 155 ...))
       (OU (153)))

(155 init_morpho
       (SI    (unif mot (conc ?x "è" ?y "er"))
      (appartient L_EE_ER ?y))          ; "céder"
       (ET_SI (nature infinitif mot)
              (groupe 1 mot)
              (classe ?c mot))
       (ALORS (egal ?z (conc ?x "è" ?y "e"))
              (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                            (personne 1)(classe ?c))
                            ?z)
              (egal ?w (conc ?x "è" ?y "es"))
              (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                            (personne 2)(classe ?c))
                           ?w)
              (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                            (personne 3)(classe ?c))
                           ?z)))

(154 init_morpho
       (SI    (unif mot (conc ?x "e" ?y "er"))
              (appartient L_E_ER ?y))          ; "achever"
       (ET_SI (nature infinitif mot)
              (groupe 1 mot)(classe ?c mot))
       (ALORS (egal ?y (conc ?x "è" ?y "e"))
              (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                            (personne 1)(classe ?c))
                            ?y)
              (egal ?z (conc ?x "è" ?y "es"))
              (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                            (personne 2)(classe ?c))
                           ?z)
```

```
            (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                          (personne 3)(classe ?c))
                         ?y))
      (SAUF (156 ...)))

 (156 init_morpho
      (SI    (unif mot (conc ?x  "eter")))        ; "jeter"
      (ET_SI (nature infinitif mot)
             (groupe 1 mot)(classe ?c mot))
      (ALORS (egal ?y (conc ?x "ette"))
             (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                           (personne 1)(classe ?c))
                           ?y)
             (egal ?z (conc ?x "ettes"))
             (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                           (personne 2)(classe ?c))
                          ?z)
             (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                           (personne 3)(classe ?c))
                          ?y))
       (SAUF (160)))

(190 init_morpho
     (SI (unif mot (conc ?x "er")))
     (ET_SI (nature infinitif mot)
            (groupe 1 mot)(classe ?c mot))
     (ALORS (egal ?y (conc ?x "é"))
            (ajouter_hyp ((nature pp)(nombre sing)(genre masc)(classe ?c))
                         ?y)
            (egal ?z (conc ?x "ant"))
             (ajouter_hyp ((nature ppresent)(classe ?c))
                         ?z)))

(165 init_morpho
      (SI    (unif mot (conc ?x "er")))         ; "chanter"
      (ET_SI (nature infinitif mot)
             (groupe 1 mot)(classe ?c mot))
      (ALORS (egal ?y (conc ?x "ons"))
             (ajouter_hyp ((nature verbe)(nombre plur)(temps present)
                           (personne 1)(classe ?c))
                           ?y)
     (egal ?z (conc ?x "ez"))
             (ajouter_hyp ((nature verbe)(nombre plur)(temps present)
                           (personne 2)(classe ?c))
                          ?z)
     (egal ?w (conc ?x "ent"))
             (ajouter_hyp ((nature verbe)(nombre plur)(temps present)
                           (personne 3)(classe ?c))
                          ?w))
      (SAUF (167 168 169 170 174 175))
      (OU (176)))
```

```
(1053 init_morpho
        (SI  (unif mot (conc ?x "dre")))
        (ET_SI  (nature infinitif mot)
                (groupe 3 mot)
                (classe ?c))
        (ALORS (egal ?y (conc ?x "ds"))
                (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                              (personne 1)(classe ?c))
                             ?y)
                (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                              (personne 2)(classe ?c))
                             ?y)
                (egal ?z (conc ?x "d"))
                (ajouter_hyp ((nature verbe)(nombre sing)(temps present)
                              (personne 3)(classe ?c))
                             ?z)))

(1471  init_morpho
        (SI (unif mot (conc ?x "clure")))
        (ET_SI  (nature infinitif mot)
                (groupe 3 mot)
                (classe ?c))
        (ALORS  (egal ?y (conc ?x "clu"))
                (ajouter_hyp ((nature pp)(nombre sing)(genre masc)(classe ?c))
                ?y))
        (SAUF  (1483)))

 (1483 init
        (SI (unif mot "inclure"))
        (ET_SI (classe ?c))
        (ALORS  (egal ?y "inclus")
                (ajouter_hyp ((nature pp)(nombre sing)(genre masc)(classe ?c))
                             ?y)))
```

Annexe 3 : Exemples de règles inverses et automate morphologique associé

Nous fournisssons ci-dessous un ensemble de règles inverses nécessaires en particulier pour le traitement des exemples présentés dans les chapitres précédents.

Nous rappelons qu'une règle inverse renvoie systématiquement à la règle la plus générale qui convient pour la désinence donnée (par exemple pour un mot terminé par *"-als"* on renvoie aux règles générales sur le pluriel des noms et adjectifs).

Lorsque la clause *Sauf* ou la clause *Si* est omise, elle est vide.

5.10 Exemples de règles traitant les mots terminés pas "s"

Règle renvoyant aux règles morphologiques sur la construction du pluriel des noms

```
(490 morpho
       (SI    (unif mot (conc ?x "s")))
       (ET_SI)
       (ALORS (regles_init (50) ?x)))
```

Règle renvoyant aux règles morphologiques sur la construction du pluriel des adjectifs

```
(491 morpho
       (SI    (unif mot (conc ?x "s")))
       (ET_SI)
       (ALORS (regles_init (70) ?x)))
```

Règle renvoyant aux règles morphologiques sur la construction du pluriel des participes passés

```
(492 morpho
       (SI    (unif mot (conc ?x "s")))
       (ET_SI)
       (ALORS (regles_init (80) ?x)))
```

Exemples de règles renvoyant aux règles morphologiques sur la conjugaison des verbes

```
(451 morpho
       (SI    (unif mot (conc ?x "es")))
       (ET_SI (egal ?y (conc ?x "er")))
       (ALORS (regles_init (151) ?y))
       (OU (454 ... )))

(454 morpho ; "achèves" "achètes" "cèles"
       (SI    (unif mot (conc ?x "è" ?y "es"))
              (appartient LETTRES ?y))
       (ET_SI (egal ?z (conc ?x "e" ?y "er")))
       (ALORS (regles_init (154 160 161) ?z))
       (OU (455)))

(455 morpho ; "cèdes"
       (SI    (unif mot (conc ?x "è" ?y  "es"))
              (appartient LETTRES ?y))
       (ET_SI (egal ?z (conc ?x "é" ?y "er")))
       (ALORS (regles_init (155) ?z)))
```

5.11 Exemples de règles traitant les mots terminés par "x"

Exemples de règles renvoyant à la construction du pluriel des noms

```
(324 morpho
       (SI    (unif mot (conc ?x "aux")))
       (ET_SI (egal ?d (les L_AUX sing "aux"))
              (egal ?y (conc ?x ?d)))
       (ALORS (regles_init (les L_AUX ?d regles_init "aux") ?y)))

(325 morpho
       (SI    (unif mot (conc ?x "eux")))
       (ET_SI (egal ?y (conc ?x "eu")))
       (ALORS  (regles_init (56) ?y)))
```

L_AUX = '(("aux" (sing "ail" "au" "al")(regles_init ("ail" 52) ("au" 51)("al" 59)))))

Exemples de règles renvoyant à la construction du pluriel des adjectifs

```
(320 morpho
       (SI    (unif mot (conc ?x "aux")))
       (ET_SI (egal ?d (les L_AUX_ADJ sing "aux"))
      (egal ?y (conc ?x ?d)))
       (ALORS  (regles_init (les L_AUX_ADJ ?d regles_init "aux") ?y)))

(321 morpho
       (SI    (unif mot (conc ?x "eux")))
       (ET_SI (egal ?y (conc ?x "eu")))
       (ALORS  (regles_init (76) ?y)))
```

L_AUX_ADJ = '(("aux" (sing "au" "al")(regles_init ("au" 71)("al" 73)))))

5.12 Exemples de règles traitant les mots terminés pas "e"

Exemple de règles renvoyant à la construction du féminin des noms

```
(200 morpho
     (SI    (unif mot (conc ?x "e")))
     (ET_SI )
     (ALORS (regles_init (1 26) ?x))
     (OU (222 209 202 221 201 ...)))

(222 morpho
     (SI    (unif mot (conc ?x "ale")))
     (ET_SI (egal ?y (conc ?x "al")))
     (ALORS (regles_init (9) ?y)))

 (209 morpho
      (SI    (unif mot (conc ?x "ette")))
      (ET_SI (egal ?y (conc ?x "et")))
      (ALORS (regles_init (7) ?y)))

(221 morpho
     (SI    (unif mot (conc ?x "se")))
     (ET_SI (egal ?y (conc ?x "x")))
     (ALORS (regles_init (2) ?y))
     (OU   ( ... )))

(202 morpho
     (SI    (unif mot (conc ?x "onne")))
     (ET_SI (egal ?y (conc ?x "on")))
     (ALORS (regles_init (6 28) ?y)))
```

```
L_TER_E = '(( "ve" (masc "f" "p"))
           ("ère" (masc "er"))
           ("euse" (masc "eur"))
           ("gue" (masc "g"))))
```

Exemples de règles renvoyant aux règles morphologiques sur la construction du pluriel des adjectifs

```
(240 morpho
     (SI    (unif mot (conc ?x "e")))
     (ET_SI )
     (ALORS (regles_init (101 126) ?x))
     (OU (252 242 251 249 241 259 ...)))

(252 morpho
     (SI    (unif mot (conc ?x "ale")))
     (ET_SI (egal ?y (conc ?x "al")))
     (ALORS (regles_init (109) ?y)))

 (249 morpho
      (SI    (unif mot (conc ?x "ette")))
      (ET_SI (egal ?y (conc ?x "et")))
      (ALORS (regles_init (107) ?y)))
```

```
(251 morpho
     (SI    (unif mot (conc ?x "se")))
     (ET_SI (egal ?y (conc ?x "x")))
     (ALORS (regles_init (102) ?y))
     (OU    (258 ...)))

(258 morpho
     (SI    (appartient L_SS_IRR_ADJ_FEM mot))
     (ET_SI (egal ?y (les L_SS_IRR_ADJ_FEM masc mot)))
     (ALORS (regles_init (les L_SS_IRR_ADJ_FEM regles_init mot) ?y)))

(259 morpho
     (SI    (appartient L_ET_FEM mot)
            (conc ?x "ète"))
     (ET_SI (egal ?y (conc ?x "et")))
     (ALORS (regles_init (118) ?y)))

(242 morpho
     (SI    (unif mot (conc ?x "onne")))
     (ET_SI (egal ?y (conc ?x "on")))
     (ALORS (regles_init (106 128) ?y)))

(241 morpho
     (SI    (unif mot (conc ?x (chaque ?d L_TER_E))))
     (ET_SI (egal ?y (conc ?x (les L_TER_E masc ?d ))))
     (ALORS (regles_init (102))))
```

```
L_SS_IRR_ADJ_FEM  =  '(("métisse" (masc "métis")(regles_init 119))
                  =  ("exprèsse" (masc "exprès")(regles_init 119))
                  =  ("épaisse" (masc "épais")(regles_init 119))
                  =  ("bêtasse" (masc "bêta")(regles_init 114))
                  =  ("fausse" (masc "faux")(regles_init 111))
                  =  ("rousse" (masc "roux")(regles_init 111))))
L_TER_E           =  '(( "ve" (masc "f" "p"))
                       ("ère" (masc "er"))
                       ("euse" (masc "eur"))
                       ("gue" (masc "g"))))
L_ET_FEM          =  '("complète" "concrète" "désuète" "discrète" "incomplète" "indi-
                       crète" "inquiète" "replète" "secrète" "préfète"))
```

Exemples de règles renvoyant aux règles morphologiques sur la conjugaison des verbes

```
(213 morpho
     (SI    (unif mot (conc ?x "e")))
     (ET_SI (egal ?y (conc ?x "er")))
     (ALORS (regles_init (151) ?y))
     (OU (215 217 ...)))

(215 morpho ; "achève" "achète" "cèle"
     (SI    (unif mot (conc ?x "è" ?y "e"))
            (appartient L_E_ER ?y))
     (ET_SI (egal ?z (conc ?x "e" ?y "er")))
     (ALORS (regles_init (...) ?z)))

(217 morpho
     (SI    (unif mot (conc ?x "ette")))
     (ET_SI (egal ?y (conc ?x "eter")))
     (ALORS (regles_init (156) ?y)))
```

5.13 Autres exemples de règles

```
(358 morpho
    (SI    (unif mot (conc ?x "d")))
    (ET_SI (egal ?y (conc ?x "dre")))
    (ALORS (regles_init (1053) ?y)))

 (360 morpho
      (SI (unif mot (conc ?x "clu")))
      (ET_SI (egal ?y (conc ?x "clure")))
      (ALORS  (regles_init (1471))))

 (361 morpho
      (SI (unif mot (conc ?x "clus")))
      (ET_SI  (egal ?y "inclure"))
      (ALORS  (regles_init (1483) ?y)))
```

5.14 Automate morphologique issu des règles inverses

L'automate morphologique dont nous présentons des extraits ici a été construit à l'aide des règles inverses et de leur combinaison. Il présente pour chaque désinence caractéristique les racines et les règles morphologiques initiales auxquelles il faut renvoyer. L'indentation permet de repérer les désinences imbriquées les unes dans les autres et qui amènent à proposer plusieurs hypothèses.

```
-----------------------------
  ?x E
    racine         : ?x
    hypothèse      : nom feminin singulier
    regle inverse  : (200)     regles init    : (1 26)

    racine         : ?x
    hypothèse      : adj feminin singulier
    regle inverse  : (240)     regles init    : (101 126)

    racine         : ?x
    hypothèse      : pp feminin singulier
    regle inverse  : (270)     regles init    : (130)

    racine         : ?x ER
    hypothèse      : verbe 1ère et 3ème personne du singulier
    regle inverse  : (213)     regles init    : (151)
-----------------------------
       ?x ALE
         racine         : ?x AL
         hypothèse      : nom feminin singulier
         regle inverse  : (222)          regles init    : (9)

         racine         : ?x AL
         hypothèse      : adj feminin singulier
         regle inverse  : (252)          regles init    : (109)
-----------------------------
       ?x OLLE
         racine         : ?x OU
         hypothèse      : nom feminin singulier
         regle inverse  : (203)               regles init    : (5)

         racine         : ?x OU
         hypothèse      : adj feminin singulier
         regle inverse  : (243)               regles init    : (105)
-----------------------------
       ?x ONNE
         racine         : ?x ON
         hypothèse      : nom feminin singulier
         regle inverse  : (202)          regles init    : (6 28)

         racine         : ?x ON
         hypothèse      : adj feminin singulier
         regle inverse  : (242)          regles init    : (106 128)
-----------------------------
```

```
      ?x È?YE
        racine        : ?x E?YER
        hypothèse     : verbe 1ère et 3ème personne du singulier
        regle inverse : (215)         regles init   : (154 160 161)

        racine        : ?x È?YER
        hypothèse     : verbe 1ère et 3ème personne du singulier
        regle inverse : (216)         regles init   : (155)
-----------------------------
          ?x ÈRE
            racine        : ?x ER
            hypothèse     : nom feminin singulier
            regle inverse : (201)         regles init   : (2)

            racine        : ?x ER
            hypothèse     : adj feminin singulier
            regle inverse : (241)         regles init   : (102)
-----------------------------
      ?x SE
        racine        : ?x X
        hypothèse     : nom feminin singulier
        regle inverse : (221)         regles init   : (2)

        racine        : ?x X
        hypothèse     : adj feminin singulier
        regle inverse : (251)         regles init   : (102)
-----------------------------
          FAUSSE
            racine        : FAUX
            hypothèse     : adj feminin singulier
            regle inverse : (258)              regles init   : (111)
-----------------------------
     INQUIÈTE
        racine        : INQUIET
        hypothèse     : nom feminin singulier
        regle inverse : (259)         regles init   : (118)
-----------------------------
     ?x ETTE
        racine        : ?x ET
        hypothèse     : nom feminin singulier
        regle inverse : (209)         regles init   : (7)

        racine        : ?x ET
        hypothèse     : adj feminin singulier
        regle inverse : (249)         regles init   : (107)

        racine        : ?x ETER
        hypothèse     : verbe 1ère et 3ème personne du singulier
        regle inverse : (217)         regles init   : (156)

        ........
-----------------------------
```

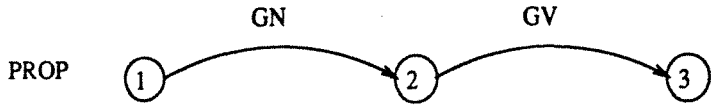
```
   ?x É
    racine        : ?x ER
    hypothèse     : pp masculin singulier
    regle inverse : (210)    regles init   : (190)
------------------------------
?x S
  racine        : ?x
  hypothèse     : nom pluriel
  regle inverse : (490)       regles init   : (50)

  racine        : ?x
  hypothèse     : adj pluriel
  regle inverse : (491)       regles init   : (70)
------------------------------
       ?x ÉS
         racine        : ?x ER
         hypothèse     :  pp masculin pluriel
         regle inverse : ((490) (210))        regles init   : ((190) (50))
------------------------------
       ?x IS
         racine        : ?x IR
         hypothèse     :  pp masculin pluriel
         regle inverse : ((490) (211))        regles init   : ((191) (50))

         racine        : ?x IR
         hypothèse     : verbe 1ère et 2ème personne du singulier
         regle inverse : (452)         regles init   : (152)
------------------------------
       ?x ES
         racine        : ?x
         hypothèse     : nom feminin pluriel
         regle inverse : ((490) (200))        regles init   : ((1 26) (80))

         racine        : ?x
         hypothèse     : adj feminin pluriel
         regle inverse : ((491) (240))        regles init   : ((101 126) (70))

         racine        : ?x
         hypothèse     : pp feminin pluriel
         regle inverse : ((492) (270))        regles init   : ((130) (50))

         racine        : ?x ER
         hypothèse     : verbe 2ème personne du singulier
         regle inverse : (451)         regles init   : (151)
------------------------------
              ?x È?yES
                racine        : ?x E?yER
                hypothèse     : verbe 2ème personne du singulier
                regle inverse : (454)    regles init   : (154 160 161)

                racine        : ?x É?yER
                hypothèse     : verbe 2ème personne du singulier
                regle inverse : (455)    regles init   : (155)
          ..........
```

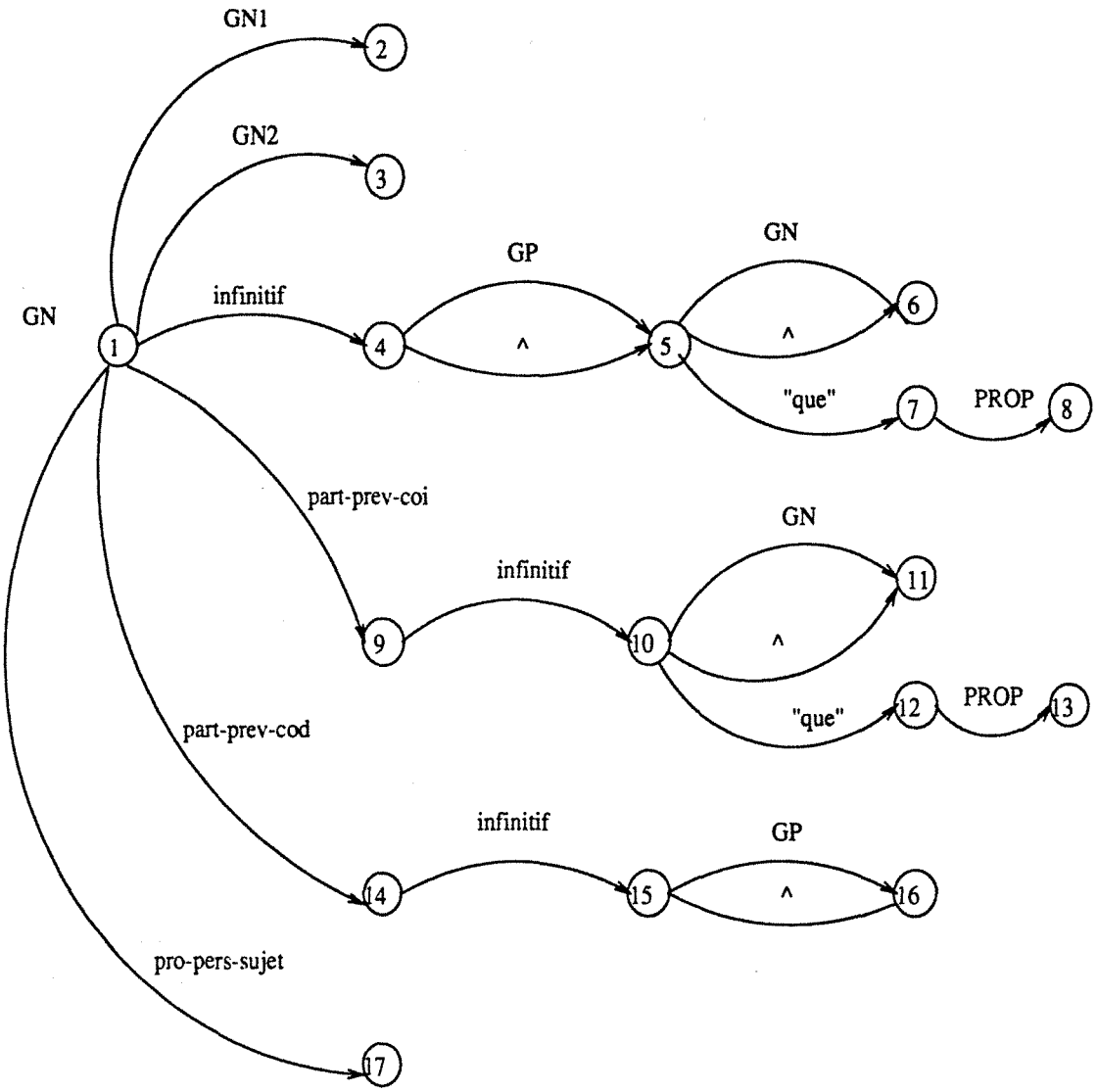
```
------------------------------
  ?x AUX
    racine         : ?x AU
    hypothèse      : nom masculin pluriel
    regle inverse  : (324)      regles init   : (51)

    racine         : ?x AL
    hypothèse      : nom masculin pluriel
    regle inverse  : (324)      regles init   : (59)

    racine         : ?x AIL
    hypothèse      : nom masculin pluriel
    regle inverse  : (324)      regles init   : (52)

    racine         : ?x AU
    hypothèse      : adj masculin pluriel
    regle inverse  : (320)      regles init   : (71)

    racine         : ?x AL
    hypothèse      : adj masculin pluriel
    regle inverse  : (320)      regles init   : (73)
------------------------------
  ?x EUX
    racine         : ?x EU
    hypothèse      : nom masculin pluriel
    regle inverse  : (325)      regles init   : (56)

    racine         : ?x EU
    hypothèse      : adj masculin pluriel
    regle inverse  : (321)      regles init   : (76)
```

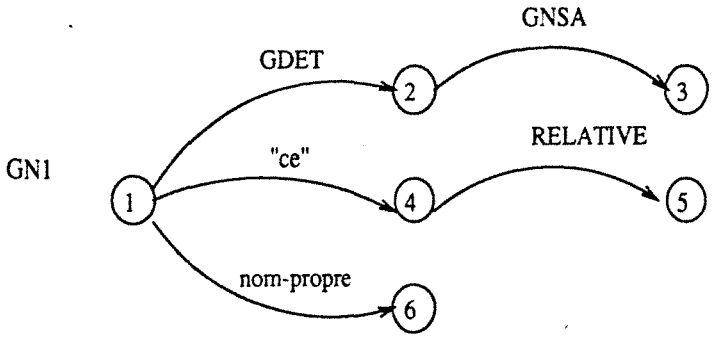
Annexe 4 : Automate de la grammaire du français utilisée pour les expérimentations

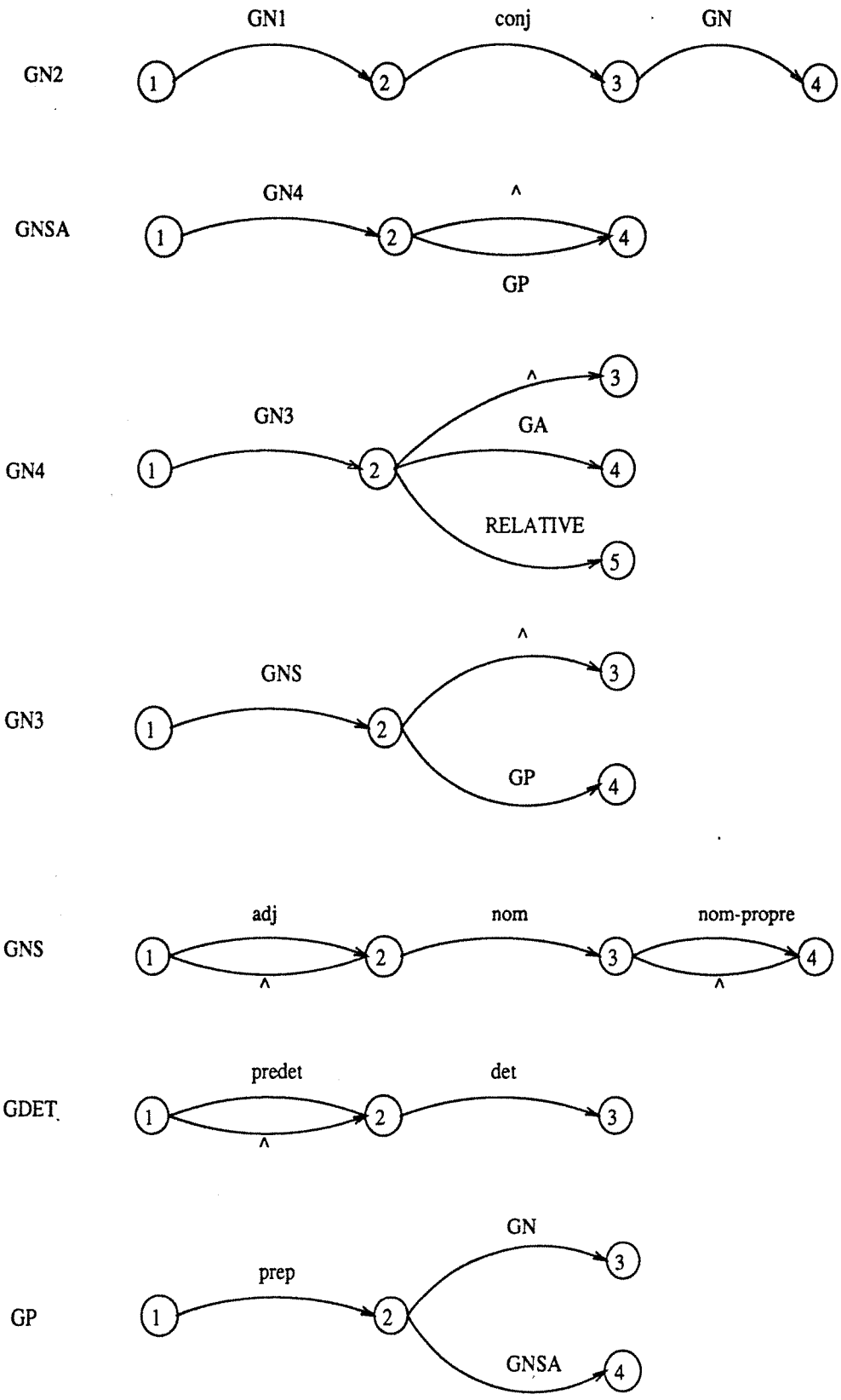
Ci-dessous est décrite sous forme d'automate récursif la grammaire du français utilisée pour les expérimentations. Les symboles en majuscules représentent les non terminaux, les symboles en minuscules entre guillements représentent des constantes, les autres représentent des symboles de catégories grammaticales de mots.

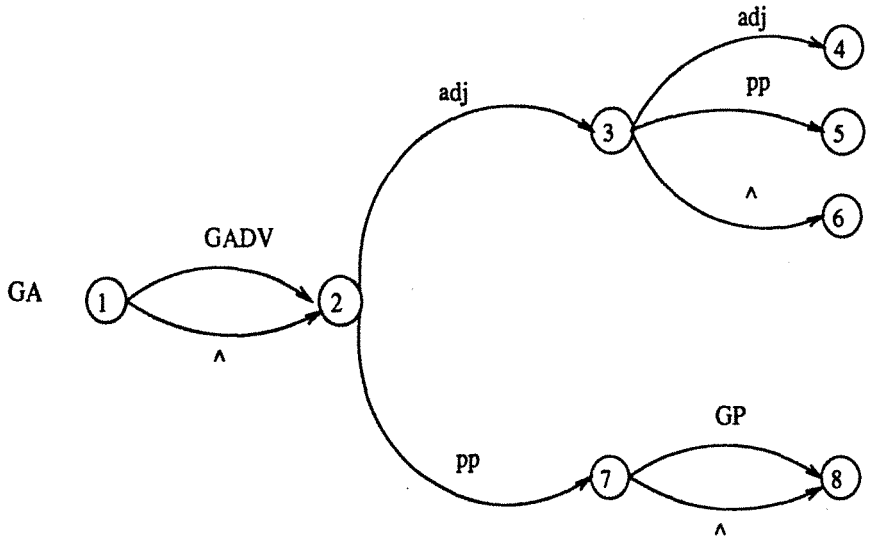
Cette grammaire est une grammaire expérimentale qui reconnaît une petite partie de la langue française. Un certain nombre de phénomènes de la langue ne sont pas considérés, l'aide d'un linguiste est indispendable si on veut concevoir et valider une grammaire pus complète.

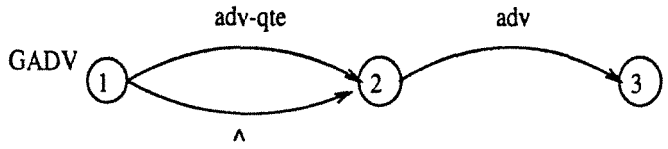

PROP
GN
GV
1
2
3

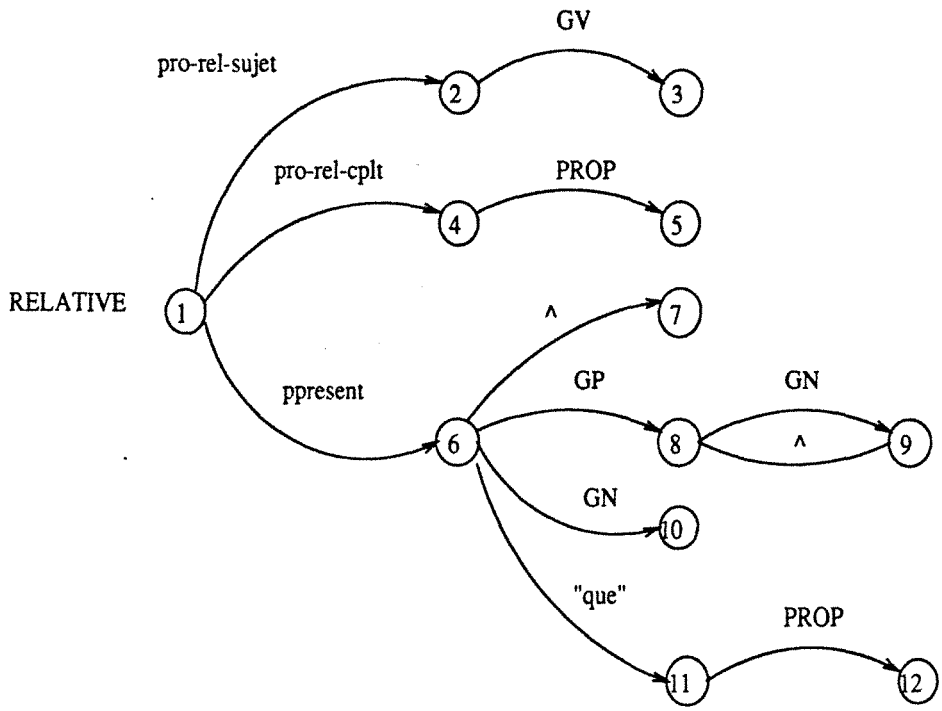

GN
GN1
GN2
infinitif
GP
GN
^
^
"que"
PROP
part-prev-coi
infinitif
GN
^
"que"
PROP
part-prev-cod
infinitif
GP
^
pro-pers-sujet
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

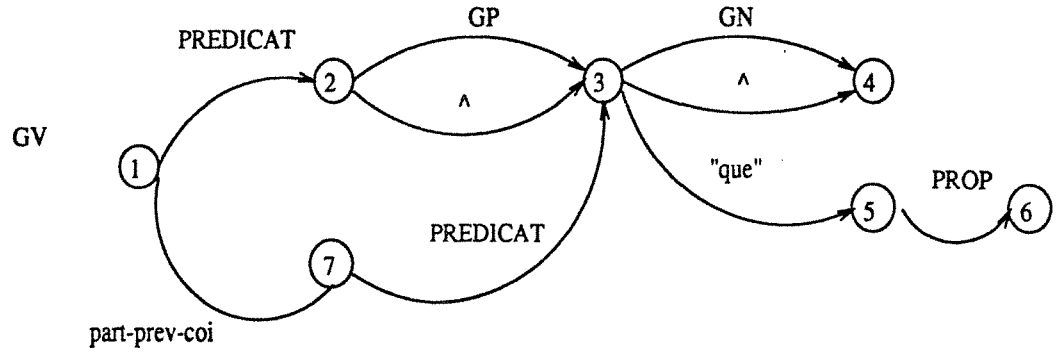

GN1
GDET
GNSA
"ce"
RELATIVE
nom-propre
1
2
3
4
5
6

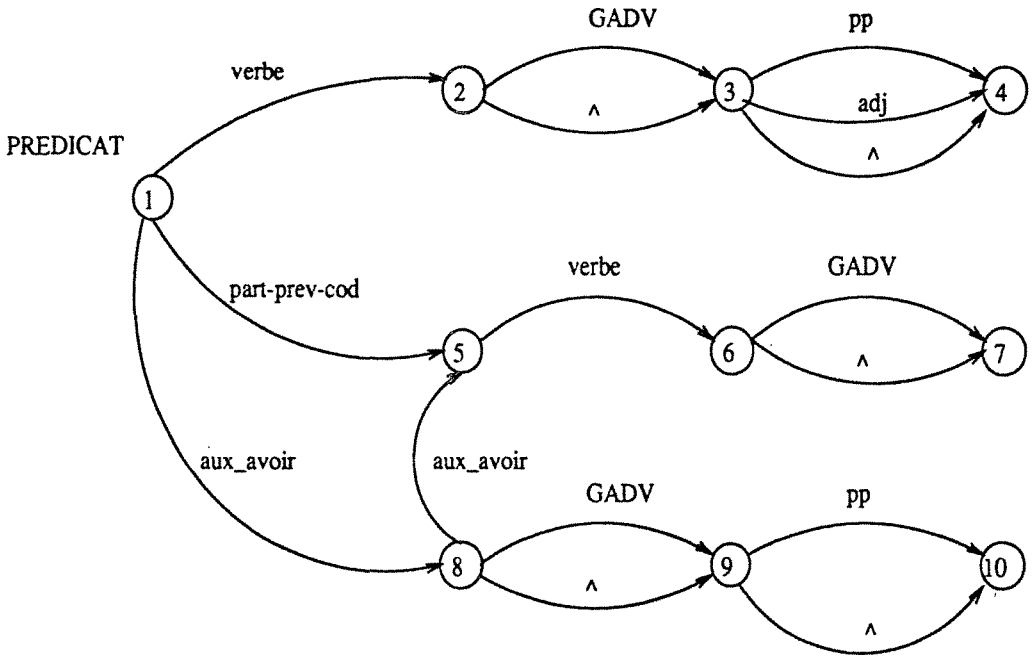

GN2
GN1
conj
GN
1
2
3
4
GNSA
GN4
^
GP
GN4
GN3
^
GA
RELATIVE
5
GN3
GNS
^
GP
GNS
adj
nom
nom-propre
GDET
predet
det
GP
prep
GN
GNSA

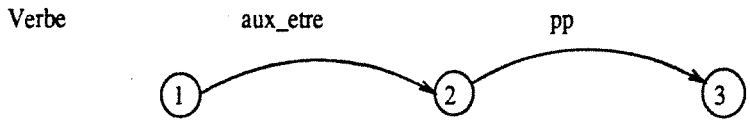

GA
GADV
adj
pp
adj
pp
GP
^
^
^
1
2
3
4
5
6
7
8


GADV
adv-qte
adv
^
1
2
3


RELATIVE
pro-rel-sujet
pro-rel-cplt
ppresent
GV
PROP
^
GP
GN
^
GN
"que"
PROP
1
2
3
4
5
6
7
8
9
10
11
12


GV
PREDICAT
GP
GN
1
2
3
4
^
^
"que"
PROP
5
6
PREDICAT
7
part-prev-coi


PREDICAT
verbe
GADV
pp
1
2
3
4
^
adj
^
part-prev-cod
verbe
GADV
5
6
7
^
aux_avoir
aux_avoir
GADV
pp
8
9
10
^
^


Verbe
aux_etre
pp
1
2
3

Annexe 5 : Exemple de règles syntaxiques

Nous fournisssons ci-dessous un ensemble de règles syntaxiques utilisées en particulier pour les exemples présentés dans les chapitres précédents.

```
(501 syntaxique
        (SI      (nature gv ?x)
                 (nature gn ?y)
                 (succ ?y ?x))
        (ET_SI   (nombre ?n ?y)
                 (personne ?p ?y)
                 (participe ?w ?x))
        (SOUS_RESERVE (nombre ?n ?x)
                      (personne ?p ?x))
        (ALORS   (egal ?z (conc ?y ?x))
                 (ajouter_hyp ((nature prop)(participe ?w))
                              ?z))
        (SAUF          (502)))


  (502 syntaxique
        (SI     (nature gv ?x)    ; accord attribut du sujet
                (nature gn ?y)
                (classe copule ?x)
                (succ ?y ?x))
        (ET_SI  (nombre ?n ?y)
                (personne ?p ?y)
                (genre ?g ?y)
                (participe ?z ?x))
        (SOUS_RESERVE (nombre ?n ?x)
                      (personne ?p ?x)
                      (nombre ?n ?z)
                      (genre ?g ?z))
        (ALORS  (egal ?w (conc ?y ?x))
                (ajouter_hyp ((nature prop)(participe ?z))
                             ?w)))
```

```
(503 syntaxique
      (SI     (nature gn1 ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x)
              (personne ?p ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature gn)(nombre ?n)(genre ?g)(personne ?p))
                           ?x)))

(505 syntaxique
      (SI     (nature gn4 ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE)
     (ALORS  (ajouter_hyp ((nature gnsa)(nombre ?n)(genre ?g))
                          ?x))
      (SAUF)
      (OU (512)))

(512 syntaxique
      (SI     (nature gn4 ?x)
              (nature gp ?y)
              (succ ?x ?y))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature gnsa)(nombre ?n)(genre ?g))
                           ?z)))

(506  syntaxique
      (SI     (nature gnsa ?x)
              (nature gdet ?y)
              (succ ?y ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE  (nombre ?n ?y)
                     (genre ?g ?y))
      (ALORS   (egal ?z (conc ?y ?x))
              (ajouter_hyp ((nature gn1)(nombre ?n)(genre ?g)(personne 3))
                           ?z)))

(511 syntaxique
      (SI     (nature pro-pers-sujet ?x))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x)
              (personne ?p ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature gn)(nombre ?n)(genre ?g)(personne ?p))
                           ?x)))
```

```
(509 syntaxique
      (SI     (nature gn3 ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature gn4)(nombre ?n)(genre ?g))
                           ?x))
      (SAUF)
      (OU  (513 560)))

(513 syntaxique
      (SI     (nature gn3 ?x)
              (nature relative ?y)
              (succ ?x ?y))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE (participe ?z ?y)
                    (nombre ?n ?z)
                    (genre ?g ?z))
      (ALORS  (egal ?w (conc ?x ?y))
              (ajouter_hyp ((nature gn4)(nombre ?n)(genre ?g))
                           ?w)))

(560 syntaxique
      (SI     (nature gn3 ?x)
              (nature ga ?y)
              (succ ?x ?y))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE (nombre ?n ?y)
                    (genre ?g ?y))
      (ALORS  (egal ?w (conc ?x ?y))
              (ajouter_hyp ((nature gn4)(nombre ?n)(genre ?g))
                           ?w)))

(515 syntaxique
      (SI     (nature gns ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature gn3)(nombre ?n)(genre ?g))
                           ?x))
      (OU  (514)))

(514 syntaxique
      (SI     (nature gns ?x)
              (nature gp ?y)
              (succ ?x ?y))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?w (conc ?x ?y))
              (ajouter_hyp ((nature gn3)(nombre ?n)(genre ?g))
                           ?w)))
```

```
(516 syntaxique
      (SI     (nature nom ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature gns)(nombre ?n)(genre ?g))
                           ?x))
      (SAUF   (517 ...)))

(517 syntaxique
      (SI     (nature nom ?x)
              (nature adj ?y)
              (succ ?y ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE (nombre ?n ?y)
                    (genre ?g ?y))
      (ALORS  (egal ?z (conc ?y ?x))
              (ajouter_hyp ((nature gns)(nombre ?n)(genre ?g))
                           ?z)))

(518  syntaxique
      (SI      (nature det ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature gdet)(nombre ?n)(genre ?g))
                           ?x))
      (SAUF  (519)))

(519  syntaxique
      (SI     (nature det ?x)   ; ''tous les ...''
              (nature predet ?y)
              (succ ?y ?x))
      (ET_SI  (nombre ?n ?x)
              (genre ?g ?x))
      (SOUS_RESERVE (nombre ?n ?y)
                    (genre ?g ?y))
      (ALORS  (egal ?z (conc ?y ?x))
              (ajouter_hyp ((nature gdet)(nombre ?n)(genre ?g))
                           ?z)))

(520 syntaxique
      (SI     (nature prep ?x)
              (nature gn ?y)
              (succ ?x ?y))
      (ET_SI (nombre ?n ?y)
              (genre ?g ?y))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature gp)(nombre ?n)(genre ?g))
                           ?z)))
```

```
(521 syntaxique
      (SI     (nature prep ?x)
              (nature gnsa ?y)
              (succ ?x ?y))
      (ET_SI  (nombre ?n ?y)
              (genre ?g ?y))
      (SOUS_RESERVE (nombre ?n ?x))
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature gp)(nombre ?n)(genre ?g))
                           ?z)))

(523 syntaxique
      (SI     (nature adj ?x))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature ga)(nombre ?n)(genre ?g))
                           ?x))
      (SAUF  (525 549 550)))

(525 syntaxique
      (SI     (nature adj ?x)
              (nature gadv ?y)
              (succ ?y ?x))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?y ?x))
              (ajouter_hyp ((nature ga)(nombre ?n)(genre ?g))
                           ?z)))

(549 syntaxique
      (SI     (nature adj ?x)
              (nature adj ?y)
              (succ ?x ?y))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x))
      (SOUS_RESERVE (genre ?g ?y)
                    (nombre ?n ?y))
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature ga)(nombre ?n)(genre ?g))
                           ?z)))

(550 syntaxique
      (SI     (nature adj ?x)
              (nature pp ?y)
              (succ ?x ?y))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x))
      (SOUS_RESERVE (genre ?g ?y)
                    (nombre ?n ?y))
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature ga)(nombre ?n)(genre ?g))
                           ?z)))
```

```
(524 syntaxique
      (SI     (nature pp ?x))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature ga)(nombre ?n)(genre ?g))
                           ?x))
      (SAUF   (526))
      (OU  (568 ...)))

(526 syntaxique
      (SI     (nature pp ?x)
              (nature gadv ?y)
              (succ ?y ?x))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?y ?x))
              (ajouter_hyp ((nature ga)(nombre ?n)(genre ?g))
                           ?z))
      (SAUF (569)))

(569 syntaxique
      (SI     (nature pp ?x)
              (nature gadv ?y)
              (nature gp ?z)
              (succ ?y ?x ?z))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?w (conc ?y ?x ?z))
              (ajouter_hyp ((nature ga)(nombre ?n)(genre ?g))
                           ?w)))

(568 syntaxique
      (SI     (nature pp ?x)
              (nature gp ?y)
              (succ ?x ?y))
      (ET_SI  (genre ?g ?x)
              (nombre ?n ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature ga)(nombre ?n)(genre ?g))
                           ?z)))

(527 syntaxique
      (SI     (nature adv ?x))
      (ET_SI)
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature gadv))
                           ?x))
      (OU  (528)))
```

```
(528 syntaxique
      (SI     (nature adv-qte ?x)  ; ''beaucoup trop ...''
              (nature adv ?y)
              (succ ?x ?y))
      (ET_SI)
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature gadv))
                           ?z)))

(530 syntaxique
      (SI     (nature pro-rel-cplt ?x)
              (nature prop ?y)
              (succ ?x ?y))
      (ET_SI (participe ?p ?y))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature relative)(participe  ?p))
                           ?z)))

(533 syntaxique
      (SI     (nature predicat ?x))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x)
              (participe ?y ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature gv)(nombre ?n)(personne ?p)(participe ?y))
                           ?x))
      (SAUF (534 543 545 ...)))

(534 syntaxique
      (SI     (nature predicat ?x)
              (nature gn ?y)
              (succ ?x ?y))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x)
              (participe ?w ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature gv)(nombre ?n)(personne ?p)(participe ?w))
                           ?z)))

(543 syntaxique
      (SI     (nature predicat ?x)
              (nature gp ?y)
              (succ ?x ?y))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x)
              (participe ?w ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature gv)(nombre ?n)(personne ?p)(participe ?w))
                           ?z))
      (SAUF     (544 ...)))
```

```
(544 syntaxique
      (SI     (nature predicat ?x)
              (nature gp ?y)
              (nature gn ?z)
              (succ ?x ?y ?z))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x)
              (participe ?u ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?w (conc ?x ?y ?z))
              (ajouter_hyp ((nature gv)(nombre ?n)(personne ?p)(participe ?u))
                           ?w)))

(545 syntaxique
      (SI     (nature predicat ?x)
              (nature prop ?y)
              (succ ?x "que" ?y))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x)
              (participe ?w ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x "que" ?y))
              (ajouter_hyp ((nature gv)(nombre ?n)(personne ?p)(participe ?w))
                           ?z))
      (SAUF (...)))

(537 syntaxique
      (SI     (nature verbe ?x))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature predicat)(nombre ?n)(personne ?p))
                           ?x))
      (SAUF  (538 539 ...)))

(538 syntaxique
      (SI     (nature verbe ?x)
              (nature gadv ?y)
              (succ ?x ?y))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature predicat)(nombre ?n)(personne ?p))
                           ?z))
      (SAUF (540 571)))
```

```
(539 syntaxique
      (SI     (nature verbe ?x)
              (classe copule ?x)
              (nature pp ?y)
              (succ ?x ?y))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature predicat)(nombre ?n)(personne ?p)
                            (participe ?y)(classe copule))
                           ?z))
      (SAUF))

(562 syntaxique
      (SI     (nature aux_avoir ?x))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x))
      (SOUS_RESERVE)
      (ALORS  (ajouter_hyp ((nature predicat)(nombre ?n)(personne ?p))
                           ?x))
      (SAUF  (541 ...)))

(541 syntaxique
      (SI     (nature aux_avoir ?x)
              (nature pp ?y)
              (succ ?x ?y))
      (ET_SI (nombre ?n ?x)
              (personne ?p ?x))
      (SOUS_RESERVE)
      (ALORS  (egal ?z (conc ?x ?y))
              (ajouter_hyp ((nature predicat)(nombre ?n)(personne ?p)(participe ?y))
                           ?z))
      (SAUF (...)))
```

Résumé

La détection des fautes d'orthographe est le domaine que nous avons choisi pour étudier l'opportunité et l'intérêt d'une analyse par affinements progressifs de textes écrits. Le développement des logiciels de traitement de textes s'est accompagné d'un accroissement de confort pour l'usager, comprenant notamment l'aide à la correction orthographique. Cependant, celle-ci est limitée comme nous le montrons en traçant un état de l'art dans le domaine. Nous apportons notre contribution en proposant des solutions en particulier pour l'accord des mots au sein des syntagmes.

La complexité du français, des langues en général, et des connaissances usuelles n'incite pas à développer une analyse à profondeur fixe. Celle-ci est généralement, pour le problème posé à résoudre à un instant donné :

- soit trop caricaturale et ne prend pas en compte les informations pertinentes pour la solution recherchée ;
- soit trop fine, et n'aboutit pas du fait de la multiplicité des informations spécifiques nécessitées par le modèle de la connaissance, mais inutiles pour le cas traité.

C'est pour tenir compte de cet état de fait, que nous proposons un système de description de la langue française (aux niveaux morphologiques et syntaxiques) qui se compose :

- d'un ensemble de règles de la connaissance renvoyant à des descriptions de plus en plus particulières ;
- d'un ensemble de règles de contrôle qui proposent des stratégies pour mettre en œuvre ces règles de la connaissance, notamment en cas d'anomalie orthographique.

Enfin, nous présentons l'expérimentation que nous avons effectuée sur un échantillon de phrases erronées récoltées dans la presse, dans des articles de journaux, et des dictées d'enfants.

Mots clés : analyse progressive, règles de stratégie, correction orthographique, processus de détection d'erreurs, langage naturel.