Département de formation doctorale en informatique
UFR STMIA

École doctorale IAE + M

Analyse d'énoncés oraux pour le Dialogue Homme-Machine à l'aide de Grammaires Lexicalisées d'Arbres

THÈSE

présentée et soutenue publiquement le 12 octobre 1999

pour l'obtention du

Doctorat de l'université Henri Poincaré – Nancy 1

(spécialité informatique)

par

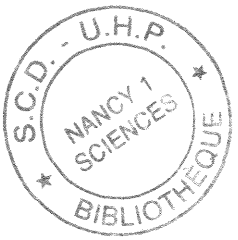
Patrice Lopez

Composition du jury

Rapporteurs :	Anne Abeillé, Maître de Conférence à l'Université Paris 7, HDR Christophe Fouqueré, Professeur à l'Université Paris 13 Jeanine Souquières, Professeur à l'Université Nancy 2
Examinateurs :	Bernard Lang, Directeur de Recherche à l'INRIA Rocquencourt Jean-Marie Pierrel, Professeur à l'Université Henri Poincaré Nancy 1 Azim Roussanaly, Maître de Conférence à l'Université Nancy 2
Invité :	Bertrand Gaiffe, Chargé de Recherche au CNRS

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Table des matières


S.C.D. - U.H.P.
NANCY 1
SCIENCES
BIBLIOTHÈQUE

Introduction

Il est étonnant de constater à quel point des choses pouvant sembler tout à fait naturelles et communes tant elles font parties de notre quotidien, apparaissent en fait d'une stupéfiante élaboration et perfection. Le regard du non-blasé peut s'émerveiller devant le langage humain. Cette faculté se développe si naturellement et rapidement durant l'enfance qu'on oublie l'incroyable sophistication et maîtrise qu'elle impose.

On a, en revanche, autour de nous, tôt fait de s'étonner des progrès des systèmes informatiques, placés comme un summum de l'évolution technologique et culturelle. La métaphore de l'ordinateur offre un nouveau cadre où tout se réinterprète, de l'intelligence humaine [Newell et al.58] [Minsky85], aux structures sociales [Simon52]. Jadis simple outil de calcul, l'ordinateur s'impose aujourd'hui comme outil de communication. Les technologies de la communication semblent nous ouvrir de nouveaux et vastes champs inexplorés, et la communication s'affirme comme une valeur universelle [Luhan68] [Sfez90] à partir de laquelle l'homme semble tendre vers le communiquer toujours plus, toujours plus vite et plus loin, avec toujours plus d'outils et d'intermédiaires. À l'inverse de cette logique, le point de vue que nous défendons est qu'au contraire l'idéal vers lequel nous devrions tendre est la minimisation de l'information, et, en nous déchargeant sur elles, ces nouvelles technologies doivent être destinées à simplifier et à réduire notre dépendance face à les flux d'informations. Il s'agit donc de développer des outils capables de nous apporter les informations pertinentes en fonction de nos activités et de nos besoins dans un monde toujours plus complexe. Suivant cet objectif à la fois de convivialité et d'efficacité, cette communication se doit d'être la plus naturelle possible, c'est-à-dire en langue naturelle.

Quelles que soient leurs prétentions, les développements de ces technologies de la communication se sont accompagnés d'un constat amer. La langue reste un système d'information et d'interaction d'une efficacité incomparable, mais aussi d'une opacité déconcertante. Depuis le débat entre Descartes et la Métrie et depuis le test de Turing, la maîtrise du langage a été placée au centre des ambitions philosophiques de l'Intelligence Artificielle (IA), ambitions aujourd'hui largement révisées à la baisse et de nature plus pragmatique (créer des produits d'ingénierie). Aux motivations philosophiques et intellectuellement stimulantes des premiers pas de l'IA ont succédés des besoins technologiques d'ordre économique et culturel. Dire que les réalisations informatiques ne concernent plus les seuls informaticiens passe aujourd'hui pour une banalité. Avec l'ouverture au grand public de la micro-informatique et le développement de réseaux à l'échelle de la planète, la conception de services de qualité et surtout aisément accessibles constitue pour les années à venir un enjeu économique considérable. Nous dépendons de plus en plus de systèmes informatiques et des raisonnements qu'ils produisent dans notre vie professionnelle et quotidienne. Il faut pourtant bien reconnaître que ces systèmes restent incapables de communiquer leurs connaissances et leurs raisonnements sous une forme humainement appréhendable : l'homme se révèle fort peu adaptable pour ce qui touche à la communication des connais-

sances et des raisonnements. De plus en plus, l'ordinateur doit se comporter comme un collaborateur capable d'interactions verbales complexes [Dessales93]. Afin de communiquer, une machine doit « connaître » le monde mais il faut d'abord qu'elle « connaisse » la langue [Carre et al.91]. Ainsi, dans une certaine mesure, le développement des technologies de l'information et de la communication passera par l'étude, la compréhension et une certaine réappropriation de notre mode de communication originel, la langue parlée, ce qu'on peut encore résumer par la formule : *connaître la langue pour la traiter automatiquement.*

Le Traitement Automatique des Langues

C'est dans ce cadre et ces besoins que se développe le traitement automatique des langues (TAL) dont l'objectif est la conception de logiciels capables de traiter de façon automatique des données linguistiques, c'est-à-dire des données exprimées dans une *langue*[1] [Fuchs et al.93], ou plutôt dans certaines langues. En effet, le TAL est devenu, toute proportion gardée, un enjeu de la compétition économique et culturelle que se livrent les grandes nations industrialisées, et par conséquent ne touche que les langues des pays les plus développés. En termes d'objectif, il s'agit pour le TAL de concevoir des logiciels capables de traiter des mots, des énoncés ou des textes, ceci afin de dialoguer avec un utilisateur, de l'aider à construire un texte ou à le traduire. Afin d'y parvenir, on peut distinguer parmi les aspects importants du TAL :

- la reconnaissance automatique de la parole, qui consiste à convertir le signal de parole produit par un locuteur humain sous une forme textuelle appréhendable par une machine ;
- la compréhension automatique de la langue, qui, de manière complémentaire au point précédent, consiste à déterminer le sens d'un énoncé et, si besoin, de générer une réponse appropriée en fonction du contexte de communication.

Après une cinquantaine d'années de travaux dans le domaine du traitement de la parole et de la langue écrite, de nombreuses applications opérationnelles comme les systèmes de dictée, d'aide à la traduction automatique, de commande vocale, de recherche d'informations ou de correction orthographique illustrent les progrès réalisés dans ce domaine. Malgré la demande sociale, aucun ne permet de mettre en œuvre de réels dialogues conviviaux entre l'homme et la machine ou des traductions automatiques suffisamment fiables. De tels systèmes restent de véritables défis scientifiques et des objectifs à long terme nécessitant des connaissances de nombreuses disciplines et faisant interagir, entre autres, phonologie, linguistique, philosophie du langage et sciences cognitives.

Un carrefour pluridisciplinaire

Le TAL, également appellé *informatique linguistique*, peut être considéré comme une branche de l'informatique qui utilise la linguistique pour atteindre ses objectifs, cette relation s'établissant dans le cadre des technologies de l'Intelligence Artificielle [Rastier91].

1. Comme précisé dans [Fuchs et al.93], le terme *langue* est plus judicieux que la plus classique dénomination de *langage naturel*, car les objets des traitements automatiques auquels nous souhaitons parvenir sont des données, textes ou paroles, exprimées dans une langue particulière et non pris au sens général de la faculté de langage. De plus, si, pour un informaticien, *naturel* est pris par opposition aux langages de programmation, le terme de *langage naturel* est utilisé par les linguistes par opposition aux langues que l'homme a tenté de construire lui-même comme l'esperanto. Le terme de langue évite donc cette ambiguité.

La réalité linguistique de la langue, à la différence de la réalité psychologique, concerne les *résultats* de l'activité langagière et non l'activité elle-même. Toute science empirique se donne pour objectif la modélisation d'un ou de plusieurs phénomènes observables. Dans le cas de la linguistique, la tâche consiste à modéliser la langue et ses propriétés. On comprend alors que le TAL s'appuie avant tout sur des connaissances issues de la linguistique structuraliste (par opposition à la linguistique simplement descriptive) dont le but est d'établir un modèle formalisable de la langue. On constate cependant très vite que, pour parvenir à une telle modélisation, il faut faire appel à des connaissances d'un autre ordre que simplement linguistique. La compréhension d'un énoncé et des contraintes qui portent sur son énonciation dépend du contexte et de la finalité de la communication.

A la différence de la linguistique, les sciences cognitives ne constituent pas forcément un passage obligé pour l'artisan TAL. Le TAL se définit avant tout comme une série d'objectifs technologiques communs, donc motivés par des besoins applicatifs. Il peut être également vu comme l'occasion d'essayer de comprendre le fonctionnement de *l'activité langagière*. Les travaux présentés dans ce mémoire, en s'inscrivant clairement suivant le premier objectif, acceptent totalement leur dimension technologique plus que scientifique, la chose n'ayant à notre sens rien de dévalorisant[2]. Dans la mesure où un système est capable de simuler les résultats du comportement langagier humain, il le réalisera en adéquation ou non avec les modèles de cognition humaine, et n'en constituera pas pour autant en soit un modèle plus acceptable qu'un autre[3]. On peut considérer que s'appuyer sur les modèles cognitifs décrivant une compétence humaine donnée ou sur les observations issues de la psycholinguistique est une bonne approche, mais une démarche ignorant les sciences cognitives peut l'être tout autant[4].

Ainsi défini selon des objectifs technologiques communs, le TAL est le lieu de collaboration entre de nombreuses disciplines, que l'on peut considérer comme scientifiques ou non, et où la pertinence pratique l'emporte.

Les systèmes de Dialogue Homme-Machine

Le domaine sur lequel porte plus particulièrement le travail présenté ici est celui de la compréhension de la parole dans des Interfaces Homme-Machine (IHM). Aujourd'hui encore, les interfaces utilisateurs sont souvent négligées, et la prise en main d'une application par une personne non-informaticienne nécessite un apprentissage le plus souvent long et fastidieux. La langue orale a un rôle important à jouer dans ce cadre. Il s'agit alors, nous le verrons, d'un moyen de communication à la fois simple, naturel et efficace.

Inclure la parole comme véritable moyen d'interaction avec l'utilisateur implique toute une série de traitements qui permettront d'obtenir une communication naturelle. Par na-

2. On peut argumenter plus généralement que, si on s'en tient à des notions d'épistémologie classique, l'informatique n'est pas une science car elle n'a pas d'objet concret d'étude, simplement la notion abstraite d'information.

3. Les sciences cognitives se fondent sur la notion de traitement de l'information (prise au sens strict de « règles ») pour expliquer la pensée humaine. À supposer que cette notion ne soit pas appropriée, l'objectif ne sera jamais atteint. Par contre, et c'est important, les hypothèses émises par les sciences cognitives restent tout à fait intéressantes et exploitables dans le cadre de l'Intelligence Articifielle dans la mesure où seule la simulation d'une tâche particulière nous intéresse et où la nature procédurale des propositions en sciences cognitives se prête à une implémentation.

4. Notons que des applications en Intelligence Artificielle moderne (*Light-AI*) sont des réussites incontestables de ce type d'approche (reconnaissance d'images, prouveur de théorème, joueur d'échec, etc.) exploitant la puissance de calcul et de la mémoire des machines avec des programmes compacts et s'écartant totalement du comportement humain sur cette même tâche.

turel, nous entendons donner aux systèmes les capacités de compréhension et de verbalisation les plus compatibles possibles avec ceux de la cognition humaine [Sabah et al.97]. Comme l'indique [Pierrel97], notre ambition est de permettre à l'utilisateur de se concentrer exclusivement sur le « quoi faire » en se libérant de toute contrainte sur le « comment l'exprimer ». Concevoir de telles interfaces ne se limite bien sûr pas à l'installation, d'un côté, d'un système de reconnaissance de la parole et de l'autre d'un simple générateur vocal pour les messages d'erreur. Par exemple, déterminer les intentions de l'utilisateur relève d'un processus particulièrement complexe incluant, entre autres, une compréhension fine des messages oraux, une interprétation dépendante du contexte de l'application et des connaissances sur l'état du dialogue en cours.

Devant la complexité mise en jeu, on comprendra que la réalisation de ce type de système se heurte à un nombre important de difficultés :

- Des problèmes d'ordre perceptif : la reconnaissance multilocuteur de la parole continue présente des limites de robustesse [Pierrel97] : les méthodes markoviennes (HMM[5]) ne peuvent fournir que de vastes treillis de phonèmes ou de mots pour assurer d'y inclure la bonne solution. Elles ne mettent pas en œuvre de connaissances linguistiques ou propres à la langue parlée, tandis que les méthodes analytiques par phonèmes se basant sur des connaissances de la langue (prosodie, informations articulatoires, etc.) restent peu performantes.

- Des problèmes liés à l'oralité : la langue orale spontanée présente une grande diversité structurelle qui se concilie difficilement avec les méthodes classiques de TAL basées sur la grammaticalité. De plus, d'une manière générale, les connaissances sur la langue orale restent encore limitées.

- Des problèmes d'ordre méta-linguistique ou pragmatique au sens de [Reboul et al.98] : l'interprétation des énoncés dans un contexte de dialogue pose un certain nombre de problèmes notamment sur la résolution de la référence et des ellipses, constructions très fréquentes dans la langue orale.

- Des problèmes d'ordre discursif : les connaissances et les modélisations formelles du dialogue et de son contexte sont lacunaires.

En prenant, comme domaine d'étude, l'analyse syntaxique des énoncés des utilisateurs, nous avons en fait choisi de nous concentrer plus particulièrement sur le second point. En quelques mots, la syntaxe désigne classiquement les règles qui conditionnent la validité grammaticale ou non d'un énoncé. Le respect de ces règles permet de déduire l'aspect compositionnel intrinsèque à l'ordre des mots, c'est-à-dire que le sens d'une phrase est plus que la somme des sens des différents mots qui la composent. Ainsi, dans une phrase comme *Washoe mange une banane*, le respect des règles syntaxiques va permettre de dire que c'est *Washoe* qui réalise l'action, tandis que la *banane* est l'objet nutritif de l'activité de Washoe. Il faut souligner que les autres problèmes évoqués sont tous liés, d'une façon ou d'une autre, par la cohérence de la langue ; il n'est pas possible de s'intéresser à une problématique sans la mettre en perspective avec celle générale posée par la maîtrise de la langue. En particulier nous considérons que la combinaison de techniques de reconnaissance de la parole et de compréhension de la langue peut améliorer le potentiel mutuel de ces deux processus.

5. *Hidden Markov Models*

Objectifs de la thèse

L'analyse du langage parlé est considérée par exemple dans [Lavie96] comme encore plus délicate que celle de textes écrits. Aux problèmes rencontrés pour l'écrit comme les ambiguïtés, viennent s'ajouter les problèmes de spontanéité de la langue orale : les hésitations, reprises, énoncés très incomplets et répétitions sont autant de caractéristiques de la langue orale employée dans un contexte de dialogue homme-machine. Ceci pose un problème pour les analyseurs syntaxiques fondés sur une notion de grammaticalité stricte qui, simplement, rejette les énoncés dès qu'une agrammaticalité est détectée. Un tel fonctionnement rend la mise en œuvre de ces analyseurs non appropriée pour la parole spontanée où, loin d'être la règle, les énoncés complets et entièrement grammaticaux sont en fait presque des exceptions.

L'objectif de cette thèse est donc de définir un niveau d'analyse adapté à des énoncés comportant de nombreuses agrammaticalités potentielles. Cette analyse doit permettre de relayer des traitements de plus hauts niveaux, comme l'analyse sémantique et le traitement des références, mais également doit être suffisamment contrainte afin de limiter le nombre d'hypothèses à considérer et interdire des combinaisons de mots impossibles en sortie d'un module de reconnaissance de la parole.

L'état de l'art du domaine montre une grande variété de travaux : tout d'abord fondés sur la linguistique avec beaucoup de théories et d'approches différentes, mais aussi reposant sur des méthodes informatiques pour l'analyse automatique robuste[6]. La diversité et le nombre de ces recherches montrent qu'il s'agit d'un domaine difficile, qui résiste à des approches simples, mais également qu'il semble possible d'aboutir prochainement à des outils automatiques performants en limitant les domaines d'application. En particulier, pour des systèmes de dialogue informatif comme un système de consultation d'horaire de trains, il semble possible de limiter l'analyse en ne cherchant à analyser dans les énoncés des utilisateurs que les informations nécessaires pour faire évoluer la tâche. C'est ce qu'on appelle les méthodes sélectives, caractéristiques des approches où l'analyse syntaxique menée est très superficielle, voir inexistante. Par contre avec des formes de dialogue plus complexes comme les dialogues de commande ou d'assistance qui nous intéressent plus particulièrement, il ne sera possible de maintenir un dialogue naturel qu'à la condition de pouvoir extraire des énoncés l'ensemble des informations qu'ils véhiculent.

Présentation du plan

Dans un premier temps, nous justifions l'utilisation d'une analyse syntaxique fondée sur un formalisme grammatical évolué pour les systèmes de dialogue. Ceci passe par une étude de la langue orale d'une manière générale et plus particulièrement de celle employée en contexte de dialogue homme-machine. En effet, malgré les nombreuses distorsions de l'oral, nous observons que d'importantes parties des énoncés en langage parlé correspondent à des noyaux grammaticaux analysables avec une grammaire correspondant à de l'écrit. Centrer dans un premier temps l'analyse sur les constituants maximaux analysables grammaticalement (potentiellement l'énoncé entier) peut être un moyen efficace d'analyse de la langue spontanée. De plus, une telle solution permettrait de bénéficier des techniques et des ressources existantes employées pour l'analyse de l'écrit. Etant donné les contraintes qu'on attend de l'analyse syntaxique dans ce contexte de dialogue et des informations nécessaires

6. Comme en témoigne par exemple les cours de Steven Abney sur l'analyse syntaxique robuste durant l'école d'été ESSLLI de 1996 avec une bibliographie contenant plus d'une centaine de références.

pour des traitements de plus haut niveau, nous plaidons pour une analyse grammaticale robuste non déterministe. Il sera possible de rendre plus efficace cette analyse, notamment avec des heuristiques de recherche et des techniques adaptées à l'oral, mais son potentiel permettra de prendre en compte les ambiguïtés, les phénomènes complexes et d'extraire des énoncés un maximum d'informations (au moins les constituants et des analyses globales dans le meilleur des cas).

Nous présentons dans la deuxième partie le formalisme grammatical retenu pour la mise en oeuvre de notre analyse : celui des Grammaires Lexicalisées d'Arbre Adjoints (LTAG ou TAG plus simplement). Nous justifions ce choix premièrement par les propriétés formelles qui permettent des mises en œuvre automatiques efficaces et deuxièmement sur la base de travaux existant en linguistique et en informatique. Nous verrons en particulier que les applications de ce formalisme à l'analyse robuste et au langage oral sont rares et avant tout exploratoires. Des objectifs de généralisation des traitements et de réutilisabilité des techniques guideront la suite de notre travail.

La troisième partie de cette thèse présente un algorithme original dédié à l'analyse d'énoncés pouvant présenter de très nombreuses agrammaticalités et où l'objectif est avant tout d'obtenir les analyses partielles les plus étendues possibles. Les stratégies descendantes, certes plus efficaces pour l'analyse de l'ensemble de l'espace de recherche, ne sont pas compatibles avec les résultats attendus. Notre algorithme dit *d'analyse par connexité* procède suivant une stratégie ascendante et se base, au cours de l'analyse, sur la représentation d'îlots et non de sous-arbres correctement analysés comme les algorithmes tabulaires classiques. Il est alors possible d'obtenir des analyses partielles plus étendues que pour les algorithmes classiques ascendants, et de considérer la notion de *parcours connexe* afin de prendre en compte la topologie de l'arbre durant l'analyse. Nous présentons de manière complémentaire une technique de compaction grammaticale afin de pallier aux désavantages de la lexicalisation complète de la grammaire en termes de complexité moyenne d'analyse.

La quatrième partie met en avant les limites de l'analyse par connexité, plus généralement les limites d'une analyse grammaticale. Ces constatations se fondent sur de premiers résultats obtenus sur un corpus de dialogues retranscrits. Les phénomènes oraux non prévus par la grammaire utilisée, dont les principes sont dédiés à l'analyse de l'écrit, se révèlent être la principale source d'erreur. Nous proposons alors deux enrichissements afin de couvrir ces phénomènes :

- un enrichissement de la grammaire utilisée afin de capter les contraintes du langage oral : en effet, même dans le cas d'énonciations spontanées très fragmentaires, toutes les analyses partielles ne sont pas valides ;
- un enrichissement du mécanisme d'analyse fondé sur l'utilisation de règles de réparation additionnelles, nommées règles paradigmatiques : nous verrons que cette proposition permet de faire progresser l'analyse en cas d'échec sans pour autant augmenter la complexité des analyses correctes. Cette étude a été réalisée en collaboration avec David Roussel.

Ces différentes propositions sont également évaluées sur corpus.

La dernière partie de cette thèse présente enfin la mise en œuvre de ces propositions au sein d'un système de conception et de tests de grammaires pour des systèmes de dialogue homme-machine nommé EGAL (pour Extraction de Grammaires d'Arbres Lexicalisées). Ce système est fondé sur une méthodologie de conception de sous-langage d'application

fondée essentiellement sur deux points :

- le recueil par expérimentations de corpus de type Magicien d'Oz, que nous fixons comme approximation d'un sous-langage d'application ;
- la spécialisation d'une grammaire générale de la langue à une application donnée, selon une approche *atelier de génie linguistique*.

Nous présentons les choix d'implantation, essentiellement motivés par la volonté d'obtenir des outils génériques et ouverts, et les résultats de cette démarche et de ce système sur le corpus GOCAD. En conclusion de cette partie, nous présentons l'extension envisagée de ce système pour la prise en compte d'autres niveaux de traitement de la langue et la place des analyseurs linguistiques associés à EGAL dans un système effectif de dialogue homme-machine.

Deux annexes viennent compléter notre présentation :

- l'annexe A rapelle quelques éléments de la théorie des langages et décrit les mécanismes d'unification ;
- la seconde annexe propose une synthèse rapide des approches probabilistes en analyse syntaxique.

Première partie

Importance et spécificités de la syntaxe dans le Dialogue Homme Machine

Chapitre 1

L'oralité

« *La production de l'écriture*, écrivait le psychologue Vygotsky à propos du rapport entre l'écrit et la langue orale, *ne suit pas le processus de production de la parole. La langue écrite est une fonction linguistique à part, différente de la langue orale à la fois en termes de structure et de mode de fonctionnement. Même un développement minimal écrit requiert un haut niveau d'abstraction. Il s'agit de parole seulement en pensée, image à laquelle manque les qualités musicales, d'expressivité et d'intonation de la langue orale. Pour apprendre à écrire, il faut se désengager de l'aspect sensoriel de la parole et remplacer les mots par des représentations de mots* »[7] [Vygotsky87].

En effet, paradoxalement, une des façons d'aborder et de comprendre le fonctionnement de l'oral est de s'intéresser à notre apprentissage de l'écrit. Platon voyait l'écrit de la même façon que la majorité des éléves apprenant l'écriture, c'est-à-dire comme une visualisation de la parole, l'inscription graphique d'un discours oral. Ce n'est cependant pas entièrement le cas : alors qu'oral et écrit dépendent des mêmes règles grammaticales, notre utilisation quotidienne de la parole spontanée s'en affranchit bien souvent. Il n'est pas surprenant de constater que les trois fautes grammaticales les plus fréquentes d'éléves débutant l'apprentissage de l'écriture en langue anglaise soient les phrases incomplètes (*the fragment*), les accords sujet verbe et les insuffisances de ponctuations et de mots de liaison (*run-on*), qui sont justement des règles particulièrement relâchées à l'oral en anglais [Rice98].

Ainsi, la langue orale, par exemple en situation de dialogue homme-homme, se caractérise par une normalité particulièrement réduite en comparaison de l'écrit. Comme expliqué dans [Luzzati95], le dialogue suppose une communication directe où la norme est d'emblée considérée comme secondaire. Les énoncés parlés sont à la fois conçus et perçus au fil de leur énonciation. Un écrivain peut reformuler à loisir une phrase jusqu'à ce qu'elle lui paraisse satisfaisante et, de façon similaire, un lecteur peut relire une phrase pour essayer de mieux en saisir sa « substantifique mœlle ». En revanche, si une erreur qui a lieu dans un énoncé parlé peut être corrigée, elle ne peut pas être effacée, comme peut l'illustrer l'exemple 1 rapporté par Freud [Freud01] d'un lapsus d'un professeur d'université.

(1) « *In the case of the female genital, in spite of the tempting ... I mean, the attempted ...* ».

En conséquence, il nous paraît tout d'abord nécessaire de définir les caractéristiques et les influences particulières qui touchent à la langue orale et qui conditionnent sa forme

7. Nous avons traduit de l'anglais ce passage que l'on pourra trouver en version originale page 98-99 de [Vygotsky87].

comme son usage. Nous mettrons en évidence ces différents points par une étude contrastive avec la langue écrite. Nous verrons ensuite comment se réalisent les productions verbales cette fois en termes de syntaxe et leurs spécificités pour ce niveau de la langue par rapport à l'écrit. Les questions qui se posent alors inévitablement sont les suivantes : peut-on traiter l'oral à partir de nos modélisations de l'écrit ? Que devient la notion de grammaticalité à l'épreuve de l'usage quotidien et spontané de la parole ?

1.1 La communication par interactions verbales

Les similarités et différences entre communication orale et écrite constituent un sujet d'intérêt en linguistique et en science de l'éducation depuis plusieurs années, dans les pays anglo-saxons tout d'abord. Plus récemment, et de façon moins massive, des recherches similaires ont été menées pour le français, essentiellement autour des travaux de Claire Blanche-Benveniste et Marie-Annick Morel. Plusieurs auteurs ont décrit la relation entre ces deux modalités comme un continuum entre l'oralité pure de notre jeune enfance et la maîtrise progressive de l'écriture. Deborah Tannen en particulier propose dans [Tannen85] de voir ce continuum interactionnel en termes de compromis relatif entre implication et information. Ainsi, si la langue orale est marquée par une implication émotionnelle et sociale forte de la personne, la langue écrite impose un contrôle, une retenue, en vue de parfaire l'aspect informatif, et n'est pas une simple retranscription d'un discours parlé élaboré en silence « dans sa tête ».

Dans la mise en œuvre d'une communication orale, [KO96] distingue :

- le matériel paraverbal (prosodique et vocal) qui correspond à toutes les unités qui accompagnent les unités proprement linguistiques, et qui sont transmises par le canal auditif : intonations, pauses, intensité articulatoire, débit, prononciation, caractéristiques de la voix ;
- le matériel non verbal transmis cette fois par le canal visuel : apparence physique des participants, cinétiques lents (postures, attitudes, etc.) et cinétiques rapides (regards, gestes, etc.) ;
- enfin le matériel verbal, c'est-à-dire l'ensemble des unités relevant de la langue (unité phonologiques, lexicales, syntaxiques et sémantiques).

Ce sont ces trois facettes de la communication qui révèlent en fait l'ensemble du locuteur. L'étude de nos mécanismes de communication relève d'une analyse détaillée de l'éthologie humaine dans ses mécanismes sémiotiques.

Le matériel verbal a longtemps été considéré en linguistique traditionnelle, comme un sous-produit de la langue, dont bien des formes caractéristiques sont considérées comme des « ratés ». Par ratés, on entend ici les bafouillements, les bégaiements et les lapsus, mais également les phrases inachevées, les répétitions, reformulations, rectifications et les marques d'hésitation. Ce style de productions verbales, dites déviantes, est massivement observé dans les différents corpus de langue spontanée. D'une façon générale, le débit rapide de parole et l'improvisation qu'imposent un dialogue se concilient mal avec l'ensemble des opérations cognitives nécessaires à la production d'énoncés, le tout contribuant à ces phénomènes qu'il est difficile de maîtriser. Cependant, il convient de préciser également que pendant une conversation, la majeure partie de ces phénomènes ont une utilité au sein des interactions, par exemple rétablir le contact. D'apparence chaotique, la langue orale révèle en fait des régularités particulières, de nature spontanée et parfois inconsciente, si

bien que Dessalle affirme que la langue employée durant une conversation décontractée correspond à l'usage le plus contraint et prévisible qu'on puisse observer [Dessales93].

Nous allons résumer les autres caractéristiques essentielles de l'oralité. En premier lieu d'un point de vue *social*, nous pouvons noter qu'elle permet de réguler les interactions entre individus [Chafe85] : des conventions de politesse, de remerciement, d'excuse sont des schémas efficaces et en quelque sorte des rituels préétablis de langue parlée [KO96]. D'autre part, en employant un niveau de langue particulier, on donne une image particulière de soi et une idée de la considération que l'on porte à son interlocuteur. Le langage sec et directif employé jadis par un colonialiste pour communiquer avec un autochtone soumis au travail obligatoire contrastera avec celui plus policé employé pour dialoguer avec un autre colon. Dans le premier cas, l'usage de la parole accompagne la domination sociale de l'autochtone colonisé, alors que pour le second, il s'inscrit dans un rapport d'égalité sociale et une relation, par exemple, de séduction.

Au niveau *interactionnel*, l'oralité se caractérise principalement des trois façons suivantes :

1. En vue de maintenir l'attention d'un interlocuteur, l'oral fait appel à des mécanismes particuliers [Tannen85][Chafe85] qui peuvent être verbaux, comme les pauses et les variations prosodiques, ou non-verbaux comme certains gestes par exemple.
2. Les interlocuteurs affichent souvent une réaction immédiate aux énoncés qui leur sont adressés, qui est une combinaison le plus souvent des trois types de matériel : verbal, para-verbal et non-verbal [Redeker84][Rubin87].
3. Les clarifications et les réénonciations peuvent se faire immédiatement. Cependant l'oral ne pardonne pas dans le sens où les erreurs ont elles aussi été communiquées (lapsus). À l'inverse, l'écrit, plus sûr, peut être poli et amélioré avant d'être donné à lire [Chafe85].

Notons que du point de vue *communicatif*, l'oral présente une persistence des énoncés souvent très limitée [Chafe85] par rapport à l'écrit qui, lui, est généralement visuellement permanent, bien que ce point dépende bien entendu du support de communication.

Sémantiquement, nous avons dégagé les trois particularités suivantes :

1. Le sujet de discussion étant souvent à l'endroit et au moment où sont produits les énoncés, la langue orale emploie d'une façon très importante les informations contextuelles [Westby85] et les déictiques (ceci, ça, là, etc.). L'écrit, à l'opposé, se détache généralement du lecteur spacialement et temporellement.
2. Le niveau d'abstraction de la langue écrite est supérieur à celui de la langue orale dans la mesure où elle manipule des représentations de mots, et non les mots directement, comme expliqué dans [Vygotsky87].
3. L'écrit manipule souvent des sujets plus abstraits et moins familiers que l'oral, ceci a particulièrement été mis en évidence par les travaux de Douglas Biber [Biber86].

Ces différents points nous aident à comprendre pourquoi l'oral a été peu étudié par rapport à l'écrit. Pour des raisons de persistence, d'une part, l'oral nécessite une étape de transcription qui se révèle en soi délicate alors que l'écrit constitue un objet d'étude facilement accessible. L'analyse de données orales peut également apparaître moins intéressante que celle de l'écrit simplement parce que les sujets de discussion sont conceptuellement moins riches, plus fonctionnels et exprimés dans un style qui rappelle le brouillon.

Du point de vue lexical, la marque de l'oral est aussi tout à fait significative :

1. La langue orale se fonde sur un ensemble familier de mots (ce que l'on appelle le

lexique utile, de l'ordre de 2000 mots en français), une syntaxe et des thèmes relativement répétitifs [Rubin87], contrastant avec l'écrit où on observe une grande richesse potentielle du vocabulaire, de la syntaxe et des idées véhiculées.

2. Comme le note [Bellenger79], l'utilisation de certains mots est privilégiée à l'oral et se révèle différente à l'écrit, c'est le cas par exemple de mot comme *terrible*, *génial* ou *nul* courant à l'oral mais qui apparaîtront excessifs à l'écrit. D'autre part, on peut même considérer que l'emploi de certains mots est propre à l'oral : par exemple les abbréviations (*télé, ciné, fac, prof*, etc.) ou les mots anglais (payer *cash*, *planning*). Le vocabulaire de l'oral est incertain, parce qu'étant donné les conditions de production du langage au cours d'un dialogue, on préfère le mot qui vient immédiatement à l'esprit et aucune vérification ne peut être effectuée. C'est le cas par exemple pour des créations douteuses comme *le désendettement, le ravivage, montrable*. Les créations de mots peuvent ainsi varier selon les locuteurs (à titre d'exemple, un journaliste parlera de *dessalage* de la mer, un second de *dessalement* et un troisième de *dessalinisation*).
3. Une distorsion des unités lexicales : phénomènes de coarticulation, disparition du début d'un mot (*aphérèse*), de la fin d'un mot (*apocope*), ou de quelque chose au milieu du mot (*syncope*), *e* muet, phénomènes d'élision à la liaison (par exemple *force animale* prononcé *for-sa-ni-mal*), deux syllabes réduites à une seule (*crase*), etc.

Des distorsions s'observent également au niveau des unités syntaxiques : nous étudierons ce point plus particulièrement dans la section suivante. On peut considérer qu'une grande partie de ces spécificités ont pour rôle de faciliter la communication et représente des facteurs d'amélioration de la qualité des interactions. En comparaison, l'écrit offre un rythme de communication beaucoup plus lent. La spontanéité est perdue au profit d'énoncés complets et d'un discours plus cohérent grâce notamment à l'utilisation de marqueurs linguistiques.

Enfin, si production d'un message oral il y a, ceci implique l'intervention complexe des muscles de la phonation, ainsi que le contrôle des différents paramètres prosodiques qui constituent l'intonation : variation de hauteur, d'intensité et de durée. La prosodie permet tout d'abord de véhiculer les différentes modalités d'une séquence (déclarative, interrogative, parenthétique, exclamative). D'autre part, un lien direct relie la structuration syntaxico-sémantique et les variations prosodiques. En particulier, certaines constructions de l'oral ne peuvent être désambiguïsées au niveau interprétatif que par des informations de nature prosodique. C'est le cas bien sûr d'interrogatives ou d'impératives réalisées sur un mode direct (*tu manges ici?*). C'est aussi le cas pour certaines constructions syntaxiques que nous aborderons plus en détail dans la section suivante. Ce lien entre syntaxe et prosodie n'est cependant pas univoque et peut présenter une très importante variabilité, surtout dans le cadre de dialogues. En effet, si la prosodie est déterminée par de nombreux facteurs linguistiques (phonologique, syntaxique, sémantique), elle est également conditionnée par des critères extra-et para-linguistiques : conditions d'élocution, émotivité du sujet, constitution physiologique des organes.

1.2 Les spécificités syntaxiques de la langue orale

La plupart des recherches sur la notion de grammaire de la langue se sont concentrées sur les problèmes de *compétence linguistique* plutôt que sur les problèmes de *performance linguistique*, autrement dit il s'agissait de caractériser les énoncés perçus comme corrects plutôt que ceux effectivement utilisés. Pour s'intéresser à la syntaxe observée dans la langue

orale, il convient de se fonder sur ce que les personnes disent effectivement et non sur des critères *a priori* de correction.

Historiquement, bien que l'exercice de la parole orale soit la forme la plus naturelle de la langue, c'est par rapport à l'écrit, admis comme norme, qu'ont été généralement effectuées les descriptions syntaxiques [KO96]. Nous allons voir maintenant en quoi ces deux formes diffèrent, c'est-à-dire les différentes manifestations de l'oralité, donc du caractère oral de la langue. En réalité, ces phénomènes sont très nombreux et la recherche de l'exhaustivité, à laquelle nous ne prétendrons pas ici, peut constituer en soi une tâche très ardue. Nous présentons dans un premier temps les structures syntaxiques privilégiées par la langue orale et quelques relâchements typiques des contraintes et des règles grammaticales.

1.2.1 Structures syntaxiques privilégiées de l'oral

D'apparence chaotique, la langue orale révèle en fait des régularités particulières. Elle se fonde sur une syntaxe réduite, présentant un degré de complexité moindre que l'écrit et relativement répétitive. Cependant comme le fait remarquer [Antoine96], si simplification générale de la syntaxe il y a, les formes complexes, rares même à l'écrit, se maintiennent, les locuteurs n'hésitant pas à les employer dans des contextes particuliers. Parmi les constructions syntaxiques souvent utilisées, citons en premier lieu les énoncés *emphatiques*, et les constructions à *topique*, c'est-à-dire construites avec reprise d'un groupe nominal ou adjectival par un pronom. Dans ces constructions, l'ordre standard des mots est modifié, le plus souvent pour marquer une insistance comme pour l'exemple (2) (tiré de [BB90]) ou les exemples (3) et (4) (construction à double topique, décrit dans [Lambrecht96]).

(2) *ce qui l'intéresse, c'est le pognon*
(3) *le camion, il avance*
(4) *Jean, sa sœur, je la déteste*

Des constructions proches, mais avec des anaphores ou des cataphores élidées, sont également relativement courantes, comme pour les exemples suivants tirés de [Lambrecht96] :

(5) *(il) mérite des baffes, ce petit con*
(6) *(il est) bizarre, ce truc*

On peut également relever les formes syntaxiques et les idiomes suivants :

- les phrases construites sur les formes *c'est ca que* ; *c'est lui qui* ; *c'est à toi que*. Notons que ces expressions ne sont pas toujours accordées au pluriel (*c'est les déménageurs qui arrivent*) ;
- les phrases construites sur la tournure *il y a*, dont la syntaxe est souvent maladroite ;
- les périphrases, liées aux problèmes de dénominalisation ;
- les locutions creuses (*il va sans dire que*) et les expressions bouche-trou (*etc. etc.*, *n'est-ce pas, comme chacun sait*) ;
- les platitudes (*à mon humble avis*) ;
- les clichés (*penchons-nous maintenant sur*) ;
- les pléonasmes (*refaire encore, sortir dehors*).

Ces constructions seraient acceptées à l'écrit, bien que souvent considérées comme maladroites.

1.2.2 Relâchements grammaticaux propres à l'oral

La langue orale relâche certaines règles syntaxiques, consciencieusement suivies à l'écrit, sans pour autant que la compréhension des énoncés par les interlocuteurs et le but de la communication en soient perturbés. [Bellenger79] relève les formes syntaxiques incorrectes suivantes :

- les tournures de phrases interrogatives en *qui est-ce qui*. Notons aussi que la langue orale ignore certaines tournures interrogatives en les compensant à l'aide de la prosodie associée à l'énoncé (il est habituel, par exemple, de dire *tu viens?* au lieu de *viens-tu?*) ;
- les *solécismes*, c'est-à-dire les fautes de syntaxe révélant l'incertitude lors de la formation du message, ou un manque d'aisance dans la langue : *aller au coiffeur*, *la robe à sa sœur*, *pallier à* ;
- des erreurs de conjugaison au passé simple ou au futur, qui sont des temps moins utilisés à l'oral : *il souria*, *ils atteigneront* ;
- la concordance des temps : *il a pris ce qu'il veut* ;
- l'accord avec l'auxiliaire avoir : *les décisions que nous avons pris* ;
- l'accord des noms collectifs : *la majorité des gens est partie* ;
- les pluriels irréguliers : *les résultats finaux.*

Les exemples que nous venons d'énumérer correspondent, selon [Bellenger79], à des agrammaticalités par non respect de règles syntaxiques existantes. Le caractère agrammatical ou non d'un énoncé reste cependant quelque peu subjectif. L'exercice de la langue parlée présente également des structures syntaxiques spécifiques. Nous avons abouti à une classification de ces phénomènes en deux grands types, d'importance très variable dans les discours oraux, qui sont, d'une part, les phénomènes de bruits et de distorsions de la langue et les phénomènes de fragmentation et d'ellipses des constituants. Nous allons tout particulièrement définir ici ces différents phénomènes et soulever quelques uns des problèmes de modélisation qui les caractérisent et auxquels il nous faudra apporter des éléments de réponse.

En vue de travailler sur les énoncés effectivement employés par les locuteurs et les textes écrits réellement observés, les linguistes ont recours à des recueils d'énoncés, des corpus, qui sont habituellement relatifs à un usage particulier de la langue ou à un domaine précis. Pour illustrer cette classification, nous avons utilisé principalement deux corpus de dialogue homme-homme[8] : le corpus SNCF [Morel88], qui a pour thème les renseignements sur les voyages férovières, et le corpus CIO [Morel89], concernant les demandes de renseignements sur l'orientation universitaire.

1.2.3 Les phénomènes de bruits et distorsions

Les hésitations, pauses et interjections

Les hésitations se manifestant par une interjection ou une pause sont des distorsions qu'il est possible de caractériser avec la propriété suivante : en éliminant de l'énoncé l'éventuelle unité lexicale d'hésitation associée, on obtient des constituants complets. Comme expliqué par [KO96], ces phénomènes peuvent en fait jouer un rôle important dans un dialogue. L'hésitation marque bien évidemment un moment d'attente dans le discours, afin,

8. Dans ce chapitre, nous nous appuyons sur les parties des corpus correspondant au dialogue homme-homme et nous ne considérons pas les variations dues à un interlocuteur artificiel.

en quelque sorte, pour le locuteur de se donner le temps de la réflexion quant à la suite de son énoncé. Le locuteur comble souvent ce « moment de flottement » par une interjection comme *euh* ou *hum*, interjection qui, cependant, ne nuit en général absolument pas à la bonne compréhension de l'énoncé de la part de son interlocuteur, celui-ci semblant totalement les ignorer. [Tannenbaum et al.65] en particulier ont montré que les hésitations se produisaient souvent avant des mots peu fréquents et peu prévisibles, c'est-à-dire ceux nécessitant un effort cognitif plus important.

Les pauses silencieuses restent le plus souvent courtes, surtout pour le français, le discours oral se conciliant mal avec les temps morts trop importants : très vite une sorte de malaise s'installe qu'il faut interrompre par une prise de parole. À ce propos, [KO96] explique que le temps entre deux tours de parole peut varier fortement d'une langue à l'autre : en moyenne 0,3 secondes pour le français, il est de 0,5 secondes pour l'anglais. Ce temps peut atteindre plusieurs minutes chez certaines communautés laponnes. Ceci permet d'expliquer pourquoi un anglais, indépendamment des problèmes de langue, peut trouver les conversations en langue française confuses, avec l'impression que les participants s'interrompent sans arrêt. Ainsi, le malaise causé par un silence long au cours d'une conversation est directement lié au rythme conversationnel général, se ressentant plus ou moins vite selon la langue. En outre, il nous est par exemple assez difficile dans un discours oral improvisé d'éviter toute hésitation se manifestant par des interjections sans un effort de concentration et de maintenir une conversation contenant de nombreuses pauses longues. Ce style d'interjections est donc une façon de meubler le canal de communication afin de préserver l'attention de l'interlocuteur et de maintenir ce canal.

Bien entendu toute interjection n'est pas une pause relative à une hésitation. De même toute pause ne manifeste pas une hésitation. Des interjections comme *ben*, *he !* permettent au locuteur de structurer son discours. L'utilisation de telles interjections est cependant très variable d'un locuteur à l'autre, certains les ignorant totalement, comme l'a montré [Luzzati83].

Ainsi, bien qu'on obtienne un énoncé complet en ignorant les hésitations, ces dernières doivent être prises en compte pour aboutir à une compréhension fine des énoncés. En particulier la présence d'une marque d'hésitation est bien souvent suivie d'autres déviations syntaxiques dues à l'oral, comme par exemple les répétitions.

Les répétitions

Tout comme les hésitations, les répétitions peuvent être vues comme un moyen d'occuper le canal de communication le temps de trouver la bonne formulation de la suite du discours. Répéter les dernières paroles permet à la fois de maintenir l'attente de l'interlocuteur et de se remémorer l'endroit interrompu de l'énoncé. Comme nous l'avons indiqué et comme l'illustre l'exemple (7), une répétition est souvent introduite par une marque d'hésitation, ceci n'étant pas non plus systématique (8).

(7) *alors* **je suis** *euh* **je suis** *partagé. (CIO)*

(8) **justement** *euh* **justement elle elle envisage** *euh* **qu'il y ait** *bon* **qu'il y ait elle envisage** *bon le fait que bon y a beaucoup d'étudiants qui postulent à ce DESS de psychologie du travail. (CIO)*

La difficulté de la prise en compte des répétitions dans un modèle grammatical est liée à leur non-prédictibilité, ce phénomène étant susceptible d'intervenir n'importe quand

lors de l'énonciation comme le montre (8). Cependant, comme pour les hésitations, les répétitions s'observent de manière privilégiée lorsque l'effort cognitif associé au processus d'énonciation est important, comme par exemple lors de l'emploi de structures syntaxiques complexes (8), elles mêmes déjà délicates à modéliser.

On observe également des répétitions sur deux tours de parole, fréquemment dans les dialogues où un locuteur est renseigné par un second. La répétition ici assure la bonne marche de la communication, il s'agit d'une façon implicite de vérifier, de la part de son interlocuteur, si le renseignement indiqué a été correctement saisi, comme pour l'exemple (9).

(9) *Opératrice : lyon part-dieu alors l'arrivée lyon part-dieu à seize heures zéro deux*
Correspondant : seize heures zéro deux. (SNCF)

Dans le cas d'une répétition qui ne reprend pas exactement ce qui a été dit, on parle d'un manière générale de *reprise*, et plus particulièrement de *précision*, de *reformulation* ou de *correction*.

Les précisions

Le discours oral oblige à maintenir un débit d'énonciation relativement constant, et l'effort cognitif nécessaire à la recherche de la bonne dénomination peut amener à employer un terme dans une relative précipitation. Face à cette situation, le locuteur ne reprendra pas l'ensemble de son énoncé mais enrichira le plus souvent ce terme flou en lui adjoingnant plusieurs mots pour en préciser le sens et la portée référentielle.

(10) *Maintenant* **coupe** *euh* **enlève** *la partie qui se trouve au dessus de la surface bleue. (GOCAD)*
(11) *Euh Hmm faire pivoter l'image un peu vers le bas à peu prés euh une vingtaine de degrés. (GOCAD)*

Au cours d'un dialogue, ce style de reprise peut aussi être la conséquence d'une interaction entre les locuteurs, un premier demandant une reformulation ou une précision en vue d'éclaicir une ambiguïté particulière, comme pour l'exemple ci-dessous (12).

(12) *Opératrice : mais également vous pouvez exercer cette profession de psychologue du travail dans des cabinets de recrutement, ou dans des cabinets de formation*
Étudiant : c'est-à-dire
Opératrice : c'est-à-dire que dans des cabinets de recrutement vous exercez du recrutement. (CIO)

Les corrections ou reformulations

Ce phénomène se manifeste par deux constituants, séparés éventuellement par des marqueurs d'hésitations, où le second est sensé corriger le premier même si celui-ci est complet. Autrement dit par rapport aux précisions, une correction a pour fonction supplémentaire « d'effacer » le constituant corrigé.

(13) *alors vendredi j'en si un à seize heures vingt et une* **non pardon** *dix-sept heures quatorze. (SNCF)*

Là encore ce phénomène de reprise peut être introduit par une hésitation, une particule négative (*non*) ou encore par une marque d'excuse (*pardon*), indiquant clairement la présence d'une correction, comme pour l'exemple (13). Sans la présence d'une telle particule, il devient impossible de distinguer ce phénomène d'une précision en se basant uniquement sur des informations grammaticales. Ce besoin d'interaction entre information syntaxique et sémantique constitue la grande difficulté de la modélisation des corrections et précisions.

Les reprises avec interruption

Une reprise avec interruption se définit comme un constituant cette fois-ci interrompu, suivi d'une nouvelle version corrigée de ce constituant qu'on nomme *reparandum*. Ce type de reprise, qu'on nomme parfois *faux-départ*, permet de corriger un énoncé sans répéter les parties correctes qui précèdent le constituant interrompu.

Notons que l'interruption peut apparaître au milieu d'un syntagme mais également au milieu d'un mot comme pour l'exemple (15) où le mot *principe* n'est pas prononcé en entier, ce qui rend délicate une vue uniquement supra-lexicale de ces phénomènes.

(14) *bon alors c'est bon euh* **j'en** *si un à neuf heures quarante. (SNCF)*

(15) *oui c'est le même* **princ...** *(ipe) non celui-ci n'est que n'a qu'un numéro. (SNCF)*

Notons que ce phénomène apparaît souvent sur deux tours de parole, avec interruption de la parole d'un locuteur comme pour l'exemple (16) et une reprise « à la volée » .

(16) *Opératrice : d'accord euh alors* **vous m'av... (ez)**
Correspondant : **pour le soir hein**
Opératrice : pardon
Correspondant : le soir de préférence
Opératice : le soir et vous m'avez dit le lundi. (SNCF)

Les effacements

Ces phénomènes correspondent à un manque involontaire dans une structure syntaxique. L'interprétation d'énoncés présentant ce type de distorsion n'est, en général, pas modifiée, l'interlocuteur ne remarquant bien souvent pas cet oubli. C'est le cas par exemple de la chute du discordantiel *ne* dans les négations, celles-ci n'étant plus marquées que par le second élément[9], comme illustré par l'exemple (17).

(17) *alors ça je peux pas vous dire hein (SNCF)*

(18) *bon je pense pas qu'actuellement on accepte ces étudiants (CIO)*

Certaines formes idiomatiques sont souvent simplifiées comme *de toutes façons* où le *de* n'est pas toujours prononcé. On peut observer également, mais de façon nettement plus anecdotique, des omissions de déterminants ou de prépositions.

9. [Antoine96] relève que le taux de chute peut atteindre 90% chez certains locuteurs dans les corpus ICP (Institut de la Communication Parlée, Grenoble).

Les différents phénomènes que nous avons définis ici sont le plus souvent considérés comme agrammaticaux, c'est-à-dire ne respectant pas les règles syntaxiques couramment admises comme correctes. Nous allons nous intéresser maintenant à d'autres phénomènes que l'on peut considérer comme grammaticaux, très fréquents à l'oral spontané, mais qui apparaîssent également comme difficiles à modéliser et très souvent ignorés des grammaires traditionnelles.

1.2.4 Les phénomènes de fragmentation et d'ellipses

Les ellipses

Les ellipses ne sont pas spécifiques au langage oral. Néanmoins on constate que les grammaires traditionnelles traitent peu ce phénomène qui est plus fréquent à l'oral qu'à l'écrit. C'est une caractéristique de l'oral d'exploiter au maximum les informations contextuelles, que ce soit le contexte perceptif (l'environnement visible dans lequel sont plongés les locuteurs), le contexte culturel ou même, notamment pour l'utilisation d'ellipses, le contexte discursif.

Une ellipse peut se définir comme un manque volontaire dans la structure syntaxique d'un énoncé. Si un mécanisme de projection permet de compléter l'énoncé à partir des structures syntaxiques précédemment utilisées, on parle alors d'ellipses syntaxiques. Si la complétude de l'énoncé ne peut se réaliser qu'à l'aide du contexte d'énonciation, on parle dans ce cas d'ellipses sémantiques. De façon générale, on parle d'ellipse lorsque la suppression d'un ou plusieurs mots d'un énoncé peut se réaliser sans que sa compréhension en souffre. Ce procédé permet d'éviter des répétitions fastidieuses et un alourdissement de l'élocution.

Pour une ellipse syntaxique, le mécanisme de projection permettant de compléter l'énoncé est soumis à des contraintes syntaxiques : la structure venant compléter ce qui a été dit doit respecter les règles d'accord et des contraintes de catégorie.

Bien qu'une ellipse ne se manifeste qu'au niveau syntaxique, sa résolution dans le cas d'ellipses sémantiques, peut nécessiter des informations également de plus haut niveau. C'est par exemple le cas de l'énoncé (19) qui nécessite une projection de nature sémantique dépendante d'informations contextuelles. C'est bien entendu là que se situe toute la difficulté de la modélisation des ellipses, car si ce mécanisme de projection nous semble immédiat et transparent, l'antécédent n'est pas forcément exprimé et constitue le résultat de contraintes complexes.

(19) *L'addition s'il vous plait !*

On rencontre des ellipses aussi bien à l'oral qu'à l'écrit, mais leur nature et leur fréquence diffèrent, comme expliqué dans [Sauvage92] :

- à l'écrit, on observe essentiellement des ellipses au sein de phrases complexes, contenant plusieurs propositions dont certaines incomplètes ;
- dans les dialogues, elles apparaissent dans les constructions simples, isolées et correspondant à des réactions rapides sous forme de questions/réponses, de précision, de contradiction, etc.

Nous observons aussi bien des ellipses sur le prédicat avec une reprise du second élément d'une alternative (20), l'élément pouvant être un groupe nominal ou verbal, que des ellipses avec absence d'arguments.

(20) *Opératrice : en ergonomie en ergonomie ou en dess psychologie du travail ?*
Etudiant : ergonomie. (CIO)

Un exemple très fréquent d'ellipse dans une discussion est celui des schémas questions-réponses. La réponse est alors le plus souvent formulée sans répéter toute la structure prédicative dans laquelle elle prend place (21).

(21) *Opératrice : de quel dess parlez-vous ?*
Etudiant : toujours d'ergonomie. (CIO)

(22) *Client : et le creusot-paris à vingt et une heures cinquante ? (SNCF)*

Dans [Frey89] les ellipses apparaissant dans les dialogues oraux orientés par une tâche précise (dialogue *finalisé*) sont classées en trois catégories :

- les ellipses *intralinguistiques* qui peuvent être complétées par un antécédent énoncé précédemment comme par exemple les réponses à des questions sous la forme d'un argument isolé (21) ;
- les ellipses *situationnelles* qui font référence aux conditions d'énonciation (19) ;
- les ellipses *extralinguistiques* qui ne peuvent être comprises qu'en faisant appel aux diverses connaissances partagées par les interlocuteurs (culturelles, sociologiques, etc.), comme en (22).

On observe également des ellipses de mots ou groupes de mots peu utiles pour les dénominations et les expressions référentielles (23).

(23) *Je regarde (des gens) jouer au échec.*

L'ellipse est un phénomène contrôlé. Dans le cas d'une ellision involontaire, on parle d'effacement ou d'omission (voir section précédente).

Les incises

L'incise est un phénomène très fréquent à l'oral qui a pour rôle de réguler et assurer la bonne marche du discours. La présence de mots comme *voilà*, *oui* ou *bon* est peu contrainte syntaxiquement, mais n'est pas non plus totalement libre. Une incise peut apparaître entre deux constituants, entre un nom et une préposition qui s'y rattache et même entre un pronom relatif et le groupe verbal associé (24). Cependant, la possibilité de voir apparaître une incise entre les marques de négations *ne* et *pas* et le verbe est à rejeter.

(24) *un poste d'informatrice* **voilà** *qui* **oui** *n'existe pas encore à paris-cinq. (CIO)*

Notons que l'incise est un phénomène qui peut être considéré comme grammatical à l'écrit et qui n'est pas propre à la production de la parole. Son rôle dans les textes écrits est indéniable, par exemple dans la bonne marche des dialogues reportés (*dit-il*), ou dans un style proche de l'oral (*si l'on peut dire*).

Nous limitons donc notre définition de l'incise à l'emploi de certains mots. Lorqu'un groupe de mots ou une proposition complète apparait au sein d'une autre proposition, nous parlerons de phénomènes de dislocation. Si la proposition initiale n'est pas interrompue, nous parlerons de juxtaposition ou de parataxe.

Les parataxes

Juxtaposition et parataxe sont deux noms utilisés pour désigner un même phénomène, qui là encore n'est pas spécifique à l'oral, mais dont la fréquence d'observation en langue parlée est très importante. Une parataxe désigne la présence au sein d'un même énoncé de plusieurs propositions complètes non liées par des connecteurs. Il en résulte un style qui apparaîtra relativement haché et maladroit à l'écrit, mais tout à fait classique à l'oral.

Comme le fait remarquer [Bellenger79], l'agencement des énoncés est tout à fait caractéristique de l'oral : il est fait de segments juxtaposés le plus souvent sans mot de liaison (parataxe). Ces énoncés simplifiés en particulier sont caractéristiques du langage oral de l'enfant qui empile les séquences, comme le montre l'exemple (25). C'est le plus souvent la prosodie qui permet de compenser la disparition des mots charnières. Les indices liés à l'intonation deviennent alors essentiels à la compréhension correcte et non ambiguë de l'énoncé.

(25) *Il est tombé, il s'est relevé, il avait pas mal.*

(26) *Opératrice : [on vous donne en même temps que certains certains objectifs les conditions de scolarité] [enfin ce genre de papiers] [les cours qui vous sont donnés] [vous avez une liste euh de des différentes entreprises qui ont reçu des étudiants jusqu'à l'année précédente] (CIO)*

Dans le cas de l'exemple (26), il est difficile de déterminer le lien entre les différentes proposition de cet énoncé, s'agit-il d'une précision, d'une reformulation, d'une apposition, d'une énumération ou de deux propositions s'enchaînant au cours du dialogue et dont le lien discursif (conséquence, opposition) reste à définir ?

Incise explicative

Ce type de phénomènes peut être vu comme plusieurs propositions dont les énonciations sont imbriquées, c'est-à-dire qu'on observe par exemple le début de la réalisation d'une première proposition qui va s'interrompre pour laisser la place à une seconde, c'est seulement à la fin de l'énonciation de cette seconde proposition que se termine celle de la première, comme illustré par l'exemple (27) extrait de [BB90].

(27) *J'avais relevé dans un journal* **je crois que c'était dans le Provençal** *une photo qui avait été prise pendant qu'ils étaient sur le toit (...) (Corpus Baumettes)*

Ce phénomène peut être comparé à la mise entre parenthèses à l'écrit. Ce type d'incises d'une proposition entière au sein d'une autre constitue un véritable défi dans l'élaboration d'un modèle de la langue. Tout d'abord, il ne s'agit pas d'agrammaticalité au sens de l'écrit, de plus, comme pour les parataxes, il peut ne pas y figurer de connecteur discursif. À cela s'ajoute la difficulté de variabilité de position des incises et la nécessité de disposer de plusieurs niveaux d'information pour interpréter correctement ces phénomènes.

On peut cependant remarquer qu'une imbriquation se réalise sur un autre registre intonatif, qui se révèle différent suivant la relation de dominance ou de juxtaposition des mots ou groupes de mots.

1.2.5 L'importance des informations contextuelles

Nous avons déjà souligné ce point en évoquant les phénomènes d'ellipses, mais cette constatation est d'autant plus flagrante lorsque l'on examine la part d'implicite véhiculée dans l'oral en comparaison de l'écrit. Il est clair que l'oral se prête par nature à l'emploi de déictiques (ceci, là, etc.). Ce type de référence particulier nécessite pour sa compréhension des informations sur le contexte perceptif environnant le locuteur.

L'importance de l'exploitation des informations implicites dans le discours peut trouver un fondement et une explication dans le principe de pertinence énoncé dans [Sperber et al.86] : un énoncé est d'autant plus pertinent qu'avec peu d'information, il amène l'auditeur à enrichir ou à modifier davantage ses connaissances ou ses conceptions. Ainsi la pertinence d'un énoncé est en proportion directe du nombre de conséquences pragmatiques qu'il entraine pour l'auditeur et en proportion inverse de la complexité et de la richesse d'information qu'il contient. Autrement dit l'énonciation privilégiée, celle qui maximise la pertinence, apparaît ici comme un compromis entre coût de production et d'interprétation de l'énoncé en contexte (minimaliser l'effort) et richesse de l'apport informationnel en contexte (maximiser l'effet).

De manière générale, toute théorie qui s'attachera à l'étude des contraintes portant sur la grande variabilité potentielle de la langue parlée ne pourra être développée qu'en considérant que la langue est faite pour être parlée, qu'il n'y a de discours et de syntaxe que pour quelqu'un et que dans une situation donnée. Au niveau syntaxique, sur lequel porte plus particulièrement cette étude, il ne sera ainsi pas possible de résoudre les ellipses sans exploiter des contraintes liées au sens et au contexte d'énonciation.

1.2.6 Quelle unité d'analyse?

Avec la notion de phrase, le choix d'une unité d'analyse syntaxique semble naturel pour l'écrit. Tout comme la segmentation en mots, au plan théorique, ce choix n'en est pas moins déjà problématique. On peut considérer qu'une phrase est constituée d'une ou plusieurs propositions, la proposition étant elle-même repérée par l'occurrence d'un verbe conjugué. Or, on constate d'une part des cas d'absence de verbe dans ce qui serait intuitivement considéré comme phrase (« *Demain, une autre réunion d'équipe? Impossible!* »), et d'autre part des cas ambigus liés à certains marqueurs. On ne peut pas non plus s'appuyer sur une définition reposant uniquement sur des notions typographiques. Si on part du principe qu'une phrase est comme une séquence de mots compris entre une majuscule et un signe de ponctuation forte (le point, le point d'interrogation ou d'exclamation par exemple), on constate vite les limites d'une telle définition.

La question d'une segmentation du discours oral se pose d'une façon encore plus problématique. Comme le rappelle [Antoine96], les linguistes travaillant sur la langue parlée rejettent une segmentation des discours oraux en unités phrastiques. Ce rejet n'est pas sans conséquences importantes dans la façon de concevoir une grammaire, puisque cette notion traditionnelle de phrase permettait de prédire les groupements possibles de mots et introduisait une segmentation utile du point de vue opérationnel.

Les travaux de [Gadet89] suggèrent de se fonder sur la notion d'énoncé sans pouvoir en apporter une définiton. On considère généralement qu'un énoncé est l'unité minimale d'énonciation, ce qui n'en est pas moins flou dans la mesure où dans le cas d'*emploi en mention* tout mot peut être employé en tant qu'énoncé acceptable.

Comme expliqué dans [Colineau97], le choix du tour de parole comme unité de segmen-

tation est tout aussi difficile, puisqu'il peut correspondre à plusieurs énoncés indépendants. Si le choix de l'acte de dialogue peut se justifier comme unité de segmentation discursive, d'un point de vue syntaxique il est inexploitable puisque la détermination de l'acte suppose déjà une analyse linguistique.

Devant l'impossibilité de s'appuyer sur la notion syntaxique traditionnelle de phrase, il faudrait pour l'oral se reporter aux notions peu précises d'énoncé ou de proposition. Si on souhaite pouvoir automatiser une analyse syntaxique de tels énoncés, il nous faudra, par la suite, tenter d'apporter des éléments de réponse à ce problème. L'un des objectifs des travaux du GARS (Groupe Aixois de Recherche sur la Syntaxe), dont l'objet général d'étude est le français parlé, est justement d'élargir l'analyse grammaticale classique pour trouver des unités qui remplaceraient celle de la phrase, résolumment jugée inadéquate [BB90].

Les phénomènes de distorsions syntaxiques décrits précédemment et la mise en cause de l'existence d'un axiome d'où classiquement dériverait la phrase ou la proposition, font que l'on peut mettre en doute la notion de grammaticalité. De façon plus générale, on peut douter également de l'application des principes de l'écrit pour modéliser l'oral, ce qui reviendrait à considérer une syntaxe de l'oral autonome de l'écrit.

1.3 Peut-on traiter l'oral à partir de l'écrit ?

1.3.1 Les travaux du GARS

Blanche-Benveniste nie l'existence d'une "langue orale" fondamentalement différente de l'écrit, d'une grammaire spéciale pour le français parlé [BB90]. Son approche de la langue orale se caractérise tout d'abord par un respect des données empiriques et un rejet des jugements d'acceptabilité. D'autre part, les travaux du GARS se fondent sur une analyse des énoncés oraux en deux niveaux. Un premier niveau correspond aux constructions verbales et nominales, le second, le niveau de la *macro-syntaxe*, fait appel à des critères sémantiques et prosodiques pour expliquer la composition des constituants identifiés précédemment. Le premier niveau se compose selon deux axes de développement : l'axe *syntagmatique* peut être vu comme une analyse syntaxique grammaticale des groupes nominaux et verbaux, guidée classiquement par un mot *recteur*, où les syntagmes sont analysés dans leur successivité. Cela signifie par exemple que pour l'énoncé (28), on peut isoler successivement quatre syntagmes correspondant à une analyse syntaxique de gauche à droite ([*hier*], [*Washoe*], [*a mangé*], [*une banane*]). Ce niveau syntaxique se fonde chez Blanche-Benveniste sur les catégories grammaticalement classiques, comme le verbe, le nom et l'adjectif.

(28) *Hier Washoe a mangé une banane*

(29) *Hier hier Washoe a englout... mangé euh une banane*

(30) *mais il y avait des gens comme Gide Claudel Valéry Malraux (interview de Roland Barthes)*

Le déroulement syntaxique n'est pas, en pratique, à l'oral, aussi linéaire et se retrouve brisé en plusieurs endroits (29). Par exemple une répétition ne peut pas être interprétée comme un enchaînement de deux unités syntagmatiques, de même pour les énumérations (30). On dit alors que ces enchaînements particuliers se déroulent sur l'axe *paradigmatique*. L'exemple (31) représente ces deux axes de déroulement, horizontalement pour l'axe syntagmatique et verticalement pour l'axe paradigmatique, de gauche à droite et de haut

en bas respectivement pour le temps. L'exemple (32) illustre cette fois cette transcription pour la réalisation d'une énumération.

(31) *hier Washoe a englout(it) euh*
hier mangé une banane

(32) *mais il y avait des gens comme Gide*
Claudel
Valéry
Malraux

Le second niveau, la *macro-syntaxe*, moins contraint par l'ordre des mots, correspond à une analyse globale distributionnelle de l'énoncé, et permet de retrouver des constructions complètes en disposant les paradigmes de syntagmes dans des grilles spécifiques à l'oral. L'objectif de ce niveau d'analyse est de définir les rapports de dépendance entre les différents syntagmes, donc d'aboutir à une représentation proche de la sémantique de l'énoncé (par exemple identifier qui réalise quelle action, qui la subit, etc.).

L'approche adoptée, devant la très grande variabilité des phénomènes observés, est avant tout descriptive et inductive: il ne s'agit pas de décrire le français parlé pour illustrer une théorie linguistique existante, mais bien d'identifier les caractéristiques que doit présenter une théorie capable d'expliquer les données du français parlé. Ainsi, dans les travaux du GARS, si une construction n'est pas observée dans au moins une vingtaine de cas similaires sur corpus d'énoncés oraux, une explication ne peut être fournie, et une analyse ferme repose toujours sur un minimum de trois cents exemples attestés. Ainsi, une explication d'un phénomène n'est réalisée qu'à la lumière de ses réalisations. Les critères d'acceptabilité intuitifs sans appuis de corpus, dont le grammairien traditionnel a souvent tendance à abuser, sont ici rejetés.

La démarche procédurale associée semblerait consister en une première analyse relativement superficielle des syntagmes possibles, puis, une fois prises en compte les variations sur l'axe paradigmatique, en un classement selon l'axe syntagmatique. Autrement dit, une fois couverts les phénomènes de type répétition, correction, énumération, etc., une analyse complète de l'énoncé est possible, selon des principes très proches de l'analyse globale de l'écrit. On remarquera cependant que deux types de phénomènes, parmi ceux présentés auparavant, échappent à la notion de variation paradigmatique. Il s'agit des effacements et des ellipses, autrement dit des phénomènes qui se caractérisent par une disparition du matériel lexical et syntaxique, plutôt qu'un rajout comme pour les autres phénomènes. Ces deux types de phénomènes devront, à notre sens, reposer sur des mécanismes de traitement différents que ceux des variations sur l'axe paradigmatique.

1.3.2 Segments grammaticaux

L'idée qu'il existe des noyaux grammaticaux parmi les différents constituants sur lesquels peut reposer une analyse grammaticale se retrouve également dans un certain nombre d'autres travaux. C'est le cas par exemple, sans considération particulière à l'oralité, de la notion de périphérie de Chanod [Chanod93]. C'est aussi le cas des travaux d'Abney [Abney91], dont la motivation dans l'utilisation de segments grammaticaux est issue des études en psycholinguistique de [Gee et al.83]. Dans ces travaux, la durée de lecture orale de texte et le groupement naïf de mots sont associés à une notion de segment. Abney l'étend en considérant une analyse syntaxique délivrant des segments partiels où les éléments problématiques ne sont pas rattachés.

Si nous reprenons le modèle de [BB90], un segment grammatical correspondra à un segment sans variation paradigmatique, pouvant comporter plusieurs syntagmes rattachés. Une prise en compte *ad hoc* des phénomènes oraux amenant des variations paradigmatiques permettra de rétablir l'axe syntagmatique, et donc une analyse grammaticale. Le niveau de la *macro-syntaxe* repose sur l'identification de constituants noyaux, considérés comme unités minimales, autour desquels des éléments s'adjoignent au titre de *préfixes*, *suffixes* et *postfixes*. Observant des regroupements très compliqués, Claire Blanche-Benveniste considère qu'étudier la combinatoire des séquences serait une tâche immense dont se dégageraient plus des tendances que des règles strictes. Les relations de dépendances complexes observées à l'oral, tout comme la résolution de certaines références qui sont également décrites dans [BB90], ne relèvent pas à notre sens des compétences d'un niveau syntaxique de traitement. [BB90] fait appel en particulier à des informations de nature sémantique et prosodique pour rendre compte de ces aspects.

1.3.3 Bilan

Les principales conclusions que nous tirons des études grammaticales sur le français parlé sont, d'une part, que l'oral ne semble pas se réaliser de manière autonome par rapport à l'écrit, et, d'autre part, que de nombreux phénomènes oraux peuvent être vus comme des variations de l'axe paradigmatique, qu'il est possible de réduire de façon à permettre une analyse grammaticale fondée sur l'écrit. La notion de segment grammatical permet d'envisager un modèle procédural de la langue parlée dans le prolongement de cette approche. Elle permet aussi de répondre d'une façon satisfaisante aux limites de la notion de *phrase* pour l'analyse de la langue orale.

Chapitre 2

Langue et Dialogue Homme-Machine

En identifiant les principales caractéristiques de la langue orale et en suivant une approche où nous tentons d'exploiter les descriptions existantes de la langue écrite, nous avons identifié les phénomènes dont il nous faudra tenir compte de façon spécifique pour une analyse automatique. Nous nous intéressons cependant à un usage particulier de la langue orale, celle employée par les utilisateurs de systèmes de dialogue Homme-Machine. Il nous semble important de présenter tout d'abord ce que l'on entend par ce style de système, d'y justifier la place et l'apport de la parole dans l'interaction avec une machine. Il sera ensuite possible de caractériser plus précisemment la langue employée, et de s'interroger sur son éventuelle simplification par rapport à la langue orale en situation de dialogue homme-homme. Nous devrons cependant mettre en perspective cet emploi particulier de l'oral avec une nouvelle contrainte, celle de la reconnaissance de la parole par la machine où la notion de grammaticalité occupe une place centrale.

2.1 Les systèmes de dialogue oral

2.1.1 Objectif

Contrairement aux voisins qu'il convient de saluer le matin en allant, par exemple, travailler, on ne se retrouve généralement pas en face d'une machine sans raison. La situation de communication qui nous intéresse ici est réduite à son aspect utilitaire. La finalité des interactions entre l'homme (l'utilisateur) et la machine est la satisfaction d'un service ou d'un besoin de l'utilisateur. On parle alors de *dialogue homme-machine finalisé par la tâche* [Falzon84].

La communication homme-machine recouvre un domaine très vaste, et n'est pas loin de s'identifier avec l'utilisation même d'un ordinateur. Notre intérêt porte ici sur l'interaction avec une machine par l'intermédiaire de la parole, ce qui suppose non seulement d'utiliser un ordinateur grâce à la voix, mais également que la machine soit capable de répondre à propos, c'est-à-dire devienne un véritable interlocuteur dans la satisfaction des besoins de l'utilisateur.

L'objectif fondamental des recherches sur le dialogue homme-machine est l'adaptation de la machine à l'homme et non l'inverse. Suivant le type d'application visé, on distingue le plus souvent trois grands types de dialogue pouvant être mis en œuvre dans un contexte informatique :

- les dialogues informatifs ou de renseignement ;

- les dialogues d'assistance ;
- les dialogues de commande.

Très schématiquement, un système de dialogue homme-machine se compose alors de trois principales composantes : un module d'analyse et de compréhension des énoncés de l'utilisateur, un module de raisonnement et de gestion du dialogue et un module de génération des interventions de la machine. Ces différents modules font appels à différents types de connaissances.

2.1.2 Les différents niveaux de connaissances

On considère le plus souvent trois grands types de connaissances statiques [Luzzati95] :

- le *modèle linguistique*[10] contenant les informations linguistiques nécessaires au fonctionnement des analyseurs (systèmes de reconnaissance et de compréhension) et du système de génération de la réponse au locuteur. En effet, comme le note [Carre et al.91], il semble naturel que ces deux fonctions correspondent à des compétences identiques réversibles : un système qui ne comprendrait que des énoncés simplifiés mais produirait des phrases très élaborées paraîtrait plutôt curieux ;
- le *modèle de la tâche* sur lequel repose une représentation de l'univers de référence dans lequel évolue le système, dont on peut distinguer la composante pragmatique (description des objets de la tâche et de leurs relations relativement à l'application) et un schéma de résolution de la tâche (buts, sous-buts, plan d'action, etc.) ;
- *le modèle du dialogue* contenant la description des diverses situations de dialogue, connues *a priori*, pouvant apparaître lors de l'interaction. Cette description peut prendre diverses formes : une représentation de la structure partielle du dialogue, une classification des échanges selon leur type (assertions, questions, etc.) ou une description hiérarchique (scripts, scénarios). C'est également le modèle de dialogue qui contient les éventuelles stratégies d'assistance disponibles.

On distingue également trois types de connaissances dynamiques :

- *l'état de la tâche* qui correspond à un ensemble de faits donnant l'historique tout d'abord des tâches et des états des objets de l'application, mais également des événements ayant une conséquence sur l'application ;
- *l'historique du dialogue* mémorisant la structure du dialogue qui s'est déroulé et son contenu ;
- *le modèle de l'utilisateur*[11] représentant les connaissances sur l'usager, dont le but est de permettre à la machine de s'adapter à son locuteur.

Les liens entre ces connaissances sont résumés figure 2.1. Notons cependant que toutes ces connaissances sont souvent très fortement dépendantes les unes des autres et qu'il est très difficile de les répartir dans différents modules comme représenté sur cette figure. De même, une des difficultés majeures de la mise en œuvre de ces systèmes est le traitement des problèmes de nature extra-linguistique, comme le calcul des référents ou la résolution des ellipses, qui reposent sur une correspondance d'informations entre les trois modèles statiques. Cette correspondance en particulier apparaît peu dans le schéma 2.1 proposé par [Luzzati95]. La résolution de la référence en sortie d'un module de sémantique sera

10. [Luzzati95] emploie le terme *modèle de langage*, qui peut à notre sens amener une certaine confusion avec les *modèles de langage* considérés en reconnaissance de la parole (par exemple *n-gramme*).

11. Il convient cependant de préciser qu'en pratique le modèle de l'utilisateur reste une connaissance statique.

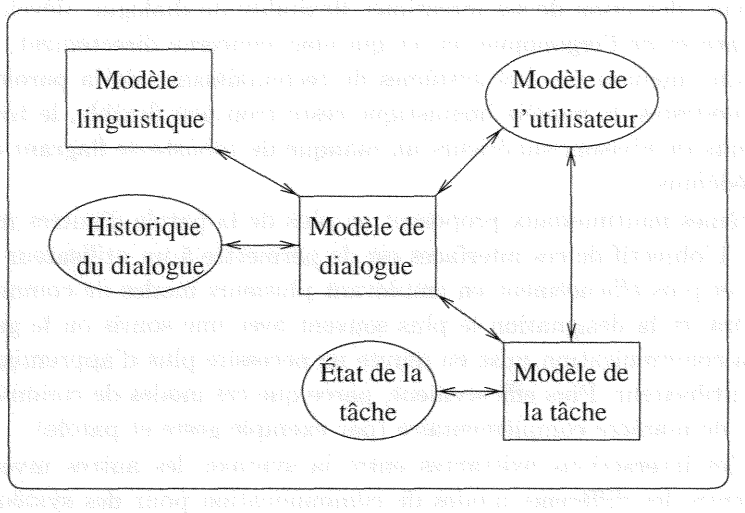


FIG. 2.1 – *Connaissances utilisées par un système de dialogue homme-machine [Luzzati95]*

obligatoirement liée d'une part au modèle de la tâche et d'autre part à l'état de la tâche [Gaiffe92]. De même, la résolution des ellipses passe par une interaction entre l'historique de dialogue et le modèle de dialogue. Ce problème soulève cette fois des questions sur la nature de l'architecture qu'il convient de mettre en œuvre. Nous verrons qu'en fait cette interrogation est récurrente quel que soit l'angle sous lequel on considère un système de dialogue. Par exemple, pour le modèle linguistique, il est nécessaire de faire interagir quatre grands niveaux d'analyse dont l'autonomie des traitements justifie la séparation :

- reconnaissance acoutico-phonétique,
- analyse morphologique et lexicale,
- syntaxe,
- sémantique de la langue.

On peut considèrer que les autres types de données relèvent de la pragmatique du discours au sens de [Reboul et al.98]. Le modèle linguistique n'en joue pas moins un rôle clef au sein d'un système de dialogue. En effet, dans la mesure où il relaie les autres niveaux de description, les performances et la robustesse des autres modules dépendent directement de celles du modèle linguistique.

2.1.3 Les limites

La plupart des systèmes de dialogue homme-machine existants permettent de mettre en œuvre une communication relativement souple, gérant de façon plus ou moins complète les erreurs ou les besoins d'aide des interlocuteurs [Chapelier96]. Des progrès restent cependant nécessaires si on veut pouvoir appliquer de telles technologies hors des laboratoires et pour des applications autres que militaires. En effet, bien que le dialogue homme-machine fasse l'objet d'un effort de recherche important depuis bientôt trente ans, les applications fiables distribuées au grand public restent peu nombreuses.

La marge de progression de ce style de système repose essentiellement sur une meilleure modélisation et une meilleure gestion du dialogue, une prise en compte plus importante

de l'utilisateur (détection de ses intentions, flexibilité du dialogue, développement de l'aspect coopératif et de l'ergonomie) et, ce qui nous concerne directement, l'amélioration de la composante linguistique. Les systèmes de reconnaissance de la parole offrent des performances modestes, le modèle linguistique reste trop peu flexible, le tout entraînant sur les traitements de niveaux supérieurs un manque de robustesse flagrant dès qu'on sort de schémas prédéfinis.

Les systèmes multimodaux proposent en plus de la parole d'autres modalités de communication. L'objectif de ces interfaces est de permettre à un utilisateur de travailler plus rapidement et plus efficacement en employant plusieurs modes de communication comme le langage oral et la désignation le plus souvent avec une souris ou le geste. Plus rapidement, car la communication mise en œuvre ne nécessite plus d'apprentissage préalable de la part de l'utilisateur. Plus efficacement, parce que ces modes de communication peuvent se combiner de manière complémentaire (par exemple geste et parole).

Décrire les interactions existantes entre la syntaxe, les autres niveaux linguistiques mais aussi entre les différents modes de communication pour des systèmes multimodaux est également un facteur de progrès potentiellement important pour ce type d'interface [Abeille et al.97].

2.2 La parole dans un contexte de DHM finalisé

2.2.1 Intérêt de la parole

Les intérêts de la parole comme modalité d'interaction sont présentés notamment dans [cAM97]. Pour résumer, l'emploi de la parole et du langage naturel est un moyen de rendre les interfaces plus simples d'emploi, la langue naturelle étant une partie intégrante de l'utilisateur acquise au fur et à mesure de son éducation et reposant sur des principes et références communs à une société. D'autre part, la parole est d'un usage immédiat, plus efficace à l'emploi qu'un clavier ou un outil de désignation. Nous nous exprimons plus rapidement à l'oral qu'à l'écrit. L'expressivité de la langue permet d'énoncer de façon très souple et très compacte des intentions complexes. Plus généralement faire reposer sur la parole l'utilisation d'une interface va dans le sens de plus de naturel, l'utilisateur pouvant se concentrer sur ce qu'il souhaite faire et non sur l'expression de commandes ou de requêtes.

Rapellons également que l'utilisation de la parole ne constitue pas toujours une bonne solution pour interagir avec un système automatique. [Husson98] présente de manière approfondie et suivant le type de tâche, les intérêts et limites d'une interface vocale, qui se révèlent souvent liés à des questions d'ergonomie.

2.2.2 Corpus d'interactions orales homme-machine

Certains corpus ont été recueillis afin d'étudier la langue orale dans le contexte de dialogue homme-homme ou homme-machine. Dans le second cas, l'expérimentation consiste en une simulation de la machine par un opérateur humain, alors appellé *compère* ou *magicien*. Ce dernier répond et exécute sur machine les actions et commandes énoncées par le sujet, qui, ignorant la simulation, s'adresse à la machine comme s'il s'agissait vraiment d'un système automatisé.

Les deux corpus auxquels nous nous sommes référés dans le chapitre précédent, corpus SNCF et CIO [12], comportent deux phases de recueil : tout d'abord dans le cadre normal

12. Ces deux corpus ont été réalisés dans le cadre du GRECO « communication parlée » du CNRS au début

d'un dialogue homme-homme, puis dans le cadre homme-machine de type magicien d'Oz. Ces deux étapes permettent de déceler les différences observées au niveau linguistique et dialogique suivant que l'utilisateur s'adresse à un homme ou à une machine.

2.2.3 Caractérisation du langage interactionnel en DHM

Notre objet d'étude, la langue employée dans un contexte de dialogue homme-machine finalisé, présente quelques particularités par rapport à la langue orale spontanée considérée en général. Les grands traits de cette spécificité tiennent en trois principaux points que nous allons développer par la suite et dont nous tenterons d'établir les conséquences pour l'analyse automatique :

- spécificité *référencielle* : l'objet du dialogue étant finalisé par la résolution d'une tâche ou par la satisfaction d'une requête dans un domaine particulier d'application, l'interprétation des énoncés et les références employés sont résolues en se restreignant aux objets et fonctionalités de l'application ;
- spécificité *langagière* : la langue et le type de dialogue sont également restreints par rapport à une communication homme-homme ;
- spécificité *sociale* : le dialogue avec une machine est nettement moins soumis à des règles et aux codes sociaux, comme la politesse, qui régulent nos interactions homme-homme quotidiennes.

Micro-monde d'application

Nous partons du principe qu'une application est restreinte à son domaine de compétence. On parle parfois de *micro-monde* qui correspond à l'ensemble d'objets, de règles et d'actions que l'application peut manipuler. L'utilisateur interagit en fait sur ce micro-monde par l'intermédiaire de la machine, qui ne peut manipuler que ses propres représentations internes. Fondamentalement, cette communication fonctionne donc selon le paradigme du *faire-faire* et non du *faire* comme de nombreuses interfaces graphiques monomodales actuelles. Lorsque l'utilisateur s'adresse à la machine, soit pour avoir accès à des renseignements contenus dans le système informatique, soit pour effectuer une action sur les objets d'une application, il fait passer une intention qu'il souhaite bien entendu voir satisfaite par la machine. Le système de dialogue se pose donc comme un intermédiaire entre l'utilisateur et le logiciel informatique proprement dit qui gère le micro-monde. L'idée que l'interprétation d'un énoncé dans un contexte de dialogue revienne à déterminer l'intention de l'utilisateur sur ce micro-monde a donné lieu à un certain nombre de travaux fondés sur la logique intentionnelle [Muller87].

Les énoncés de l'utilisateur ne feront ainsi référence qu'à des notions du domaine de l'application. Bien entendu, la conséquence de ces différents principes pour nos objectifs est une combinatoire limitée pour la résolution d'une part des références aux objets de l'application et d'autre part des références aux actions possibles.

Richesse structurelle

Dans [Carbonell83], une étude empirique des énoncés produits par des utilisateurs de systèmes de dialogue Homme-Machine simulés a montré qu'utilisateurs et compères pouvaient éviter relativement facilement l'emploi de structures syntaxiques complexes, mais

des années 80.

pouvaient très difficilement se forcer à employer des énoncés complets et donc éviter l'emploi d'anaphores. Cette étude vient confirmer l'intuition : il est indispensable pour l'analyse d'énoncés en contexte de dialogue homme-machine de concentrer ses efforts dans la modélisation des phénomènes elliptiques et anaphoriques plutôt que dans celle de phénomènes syntaxiques licites mais complexes dont l'usage se révèle inexistant ou réellement anecdotique. Mary-Annick Morel [Morel89] est parvenue à la même conclusion lors de l'analyse du corpus CIO : le compère lui-même, bien que devant rejeter les tournures elliptiques de l'utilisateur, les employait fréquemment dans ses questions et réponses.

Sous-langage d'un système de dialogue

Un sous-langage se définit comme un ensemble d'énoncés liés par un sujet limité, utilisés pour une fonction particulière et engendrés par une grammaire et un vocabulaire spécifiques [Deville89]. Deux facteurs viennent restreindre le langage général dans un système de Dialogue Homme-Machine : d'une part le type de dialogue finalisé qui est mis en œuvre (dialogue de commande, d'assistance, etc.) et d'autre part le domaine d'application du système. Un sous-langage n'est pas un simple sous-ensemble du langage complet : une application peut nécessiter l'usage des termes techniques ne relevant que du domaine. Il est donc important qu'un système propose des méthodes pour décrire de nouveaux mots et de nouveaux contextes syntaxiques afin de prendre en compte les structures échappant à une grammaire générale de l'écrit.

Du point de vue opératoire, outre la diminution de la combinatoire, une grammaire restreinte à un sous-langage rend également réaliste la possibilité de mettre en œuvre une grammaire écrite manuellement, chose nettement plus difficile par exemple pour des systèmes de dictée automatique.

Une machine qui se pose en tant que tel

L'utilisateur ne s'adresse pas à un système en le considérant comme un interlocuteur humain, ce qui induit un aspect particulier de la langue. C'est la dimension sociale dont nous parlions dans le chapitre précédent et qui trouve sa réalisation ici dans un type de dialogue proche du maître-esclave. Sachant qu'il s'adresse à une machine, l'utilisateur va à l'essentiel, c'est-à-dire à la résolution de la tâche, sans marque de politesse particulière. Par exemple, [Bilange92] relève que, sur 300 dialogues homme-machine simulés, seulement 10% d'entre eux contiennent un *bonjour* de la part de l'utilisateur dans l'ouverture du dialogue, alors que la proportion est inversée avec un interlocuteur humain.

La dimension sociale de la langue est ici le plus souvent réduite à son minimum, l'utilisateur ne souhaite pas donner une image particulière de lui-même, il emploie un langage impersonnel selon une relation de dirigeant à dirigé et le paradigme de faire-faire que nous avons introduit précédemment.

Notons cependant que d'autres types de relations peuvent être observés lors des dialogues homme-machine, alternant avec la relation dirigeant-dirigé, ou même s'y substituant totalement. C'est le cas des relations de partenariat (33) ou d'action directe, où l'application en tant « qu'interlocuteur » agissant disparaît, donnant alors lieu à une interaction relevant alors du paradigme du *faire* (34).

(33) *nous allons prendre l'armoire, que nous portons ici (MacOZ)*

(34) *je change le grand lit de place (MacOZ)*

En conséquence, si on ne souhaite pas forcer l'utilisateur à s'adapter à un type de dialogue, on ne peut pas complètement ignorer l'aspect social de la langue et supposer, par exemple, que toutes les commandes ou requêtes soient formulées à l'impératif.

2.2.4 Langage naturel plutôt que pseudo-naturel : le prix à payer

Les premières grammaires structurelles d'analyse de la langue se sont révélées inefficaces face à la complexité de la langue. Elles connurent cependant une réussite tout à fait remarquable pour l'analyse de langages formels comme les langages de programmation. Ces langages se caractérisent par un niveau syntaxique déterministe, c'est-à-dire que le résultat de l'analyse grammaticale est toujours unique. De plus, le niveau de description des unités composant les phrases à analyser est très général : le comportement syntaxique de chaque entrée lexicale étant donné par un nombre très limité de catégories syntaxiques.

Un *langage artificiel à consonance naturel* ou *langage pseudo-naturel* peut être vu comme un compromis en réponse à la complexité du langage naturel pris dans son ensemble, permettant la mise en œuvre de ces techniques bien connues, tout en employant des mots du lexique de la langue et un comportement syntaxique rappelant celui que nous employons quotidiennement bien que très stéréotypé (par exemple commande+objet+paramètre dans le système Diapason [Souvay92]). L'analyse syntaxique est donc ici univoque. Remarquons que ce type de langage n'est pas réservé exclusivement aux informaticiens. Les dialogues entièrement normés par exemple employés dans la marine militaire sont artificiels. Bien entendu ce type de langages artificiels nécessite une phase d'apprentissage plus ou moins longue de la part des utilisateurs et un effort constant afin de normaliser ses énoncés dans l'espoir d'être compris par la machine.

Pour la prise en compte effective du langage naturel, il faut tout d'abord être capable de gérer le problème de l'ambiguïté des différents niveaux de descriptions. Le niveau syntaxique est par nature ambigu.

De même, la sémantique ne permet pas toujours d'aboutir à une unique interprétation d'un énoncé ; il faut exploiter le contexte dans lequel le discours est tenu pour décider d'une interprétation. Pour le dialogue homme-machine plus particulièrement, ceci correspond au contexte de l'application. D'autres aspects de la langue naturelle viennent la distinguer de langage pseudo-naturel : la prise en compte des distorsions de la langue orale étudiées au chapitre précédent va également au-delà du potentiel déployé dans des systèmes de langage pseudo-naturel.

Un fossé sépare donc la complexité de la langue de celle des langages pseudo-naturels. Il est significatif en fait de l'erreur d'appréciation des difficultés mises en jeu dans les premiers travaux d'analyse et de compréhension de la langue.

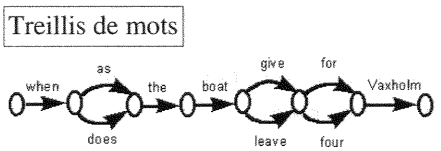
2.2.5 Importance de la grammaticalité dans le processus de reconnaissance de la parole

La dernière des principales spécificités de la langue orale employée pour interagir avec un système de dialogue est tout simplement la nécessité première de reconnaître cet énoncé. Cette contrainte en fait implique un certain nombre de conséquences quant au rôle de l'analyse mise en œuvre, qui n'existe pas par exemple pour l'analyse de la langue orale retranscrite. Tout comme l'homme reconnait plus facilement les mots énoncés par son interlocuteur grâce à la position qu'ils occupent dans les phrases, il faut pouvoir exploiter cette information pour la reconnaissance automatique de la parole. Nous commencerons

donc par présenter les techniques habituellement utilisées dans les systèmes de dialogue existants pour la reconnaissance de la parole.

Techniques de reconnaissance de la parole

Les techniques de Reconnaissance Automatique de la Parole (RAP) correspondent au niveau de traitement perceptif d'un système de dialogue homme-machine, par opposition aux traitements dits *supérieurs* des niveaux interprétatifs. Ce premier niveau a en charge l'acquisition du signal de parole et la transcription de ce signal en une séquence d'unités linguistiques, généralement une séquence de mots [Husson98]. Une première étape consiste habituellement à calculer une représentation du signal numérisé moins redondante que le signal acoustique proprement dit. Les techniques utilisées pour cette première phase de paramétrisation aboutissent à des représentations spectrales ou cepstrale du signal intégrant généralement des aspects perceptifs (MFCC[13], PLP[14]). Une seconde étape consiste en la construction des unités linguistiques afin de parvenir à une suite de mots, pour les méthodes dites *globales*, ou à une suite de phonèmes, pour les méthodes *acoustico-phonétiques*. En pratique, du fait des incertitudes de reconnaissance, le résultat fourni est plutôt une représentation compacte d'un ensemble d'hypothèses sous la forme d'un treillis soit de mots, comme celui de la figure 2.2, soit de phonèmes. Il est aussi habituel d'associer à chaque hypothèse de ce treillis un score de confiance.



Liste des N-meilleurs hypothèses

when as the boat give for Vaxholm
when does the boat give for Vaxholm
when as the boat leave for Vaxholm
when does the boat leave for Vaxholm
when as the boat give four Vaxholm
when does the boat give four Vaxholm
when as the boat leave four Vaxholm
when does the boat leave four Vaxholm

FIG. 2.2 – *Un exemple de treillis d'hypothèses de mots fourni par un système de reconnaissance de la parole (d'après [Strom97]).*

La plus importante source de difficultés dans la tâche de RAP est la grande variabilité intrinsèque au signal de parole lié à des facteurs dus aux locuteurs, au processus de production (phénomène de coarticulation), à l'environnement sonore, etc. Trois grandes approches de la reconnaissance de la parole existent : la première, dite *analytique*, tente d'exploiter des connaissances de la langue modélisée explicitement, la seconde, dite *auto-adaptative* se base sur des techniques d'apprentissage, enfin une troisième approche se fonde sur la programmation dynamique, technique de reconnaissance des formes. Au vue des systèmes industriels de RAP les plus récents, l'approche auto-adaptative, et tout particulièrement

13. *Mel Frequency Cepstral Coefficients.*
14. *Perceptually-based Linear Prediction.*

les HMMs (Hidden Markov Model), supplante actuellement les techniques fondées sur des connaissances phonétiques explicites, la troisième approche étant inopérante. Autrement dit, les recherches en reconnaissance de la parole relèvent aujourd'hui plus des départements d'ingénierie que des départements de linguistique ou linguistique informatique.

Les résultats classiques de reconnaissance de mots pour des systèmes de dialogue non triviaux, c'est-à-dire comportant au moins 200 mots de vocabulaire et dans des conditions normales d'utilisation (hors laboratoire) indiquent un taux d'erreur de l'ordre de 15% à 20%, soit une erreur par énoncé environ en considérant la meilleure hypothèse fournie par le système de RAP. Ceci signifie que la solution correcte de reconnaissance ne se trouve habituellement pas en considérant simplement la meilleure hypothèse fournie par le système, mais parmi un ensemble de 10 à 100 hypothèses de phrase (*n-best*). Les phénomènes oraux comme les hésitations, répétions, reprises, etc. peuvent être reconnus en joignant un modèle d'intonation et de durée et des modèles qui prennent en compte des formes syntaxiques. On peut considérer que la modélisation de ce style de phénomènes oraux n'en est cependant encore qu'à ses débuts [Rickley94] [Shriberg94].

Filtrer les hypothèses de reconnaissance.

La question que soulèvent ces différentes techniques est leur intégration dans un système de dialogue, c'est-à-dire avec un niveau interprétatif. Deux grandes approches se distinguent : la première dite de *post-décodage* se base sur un processus séquentiel où une première étape vise à produire un treillis d'hypothèses de mots obtenues par le système de reconnaissance selon des contraintes apportées par un modèle de langage. L'architecture de tels systèmes est illustrée par la figure 2.3. L'analyse interprétative proprement dite est menée sur la base de ce résultat et permet donc, en plus d'obtenir une représentation sémantique associée à ces énoncés, de filtrer des hypothèses agrammaticales (i.e. qui ne vérifient pas la grammaire utilisée pour l'interprétation). La seconde approche tente de réaliser une intégration plus forte des deux processus en exploitant directement, au cours du décodage acoustico-phonétique, des hypothèses de mots reconnus et les contraintes apportées par la grammaire utilisée pour l'interprétation. Nous allons maintenant étudier plus en détail les conséquences de ces deux approches pour le fonctionnement du système de dialogue.

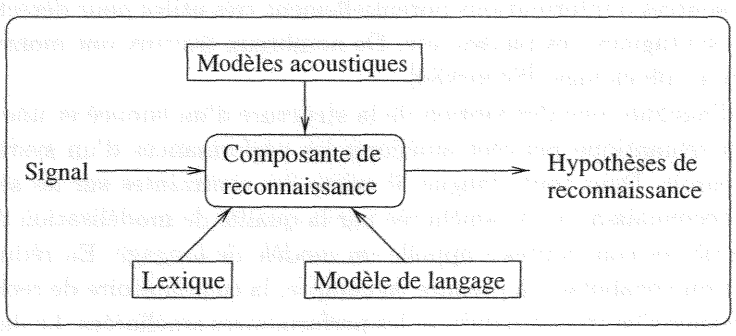


FIG. 2.3 – *Système actuel de reconnaissance [Husson98]*

Le principe de la première approche de post-décodage est de mener indépendamment les tâches de reconnaissance de la parole et d'interprétation des énoncés. La justification sous-jacente de cette approche est que les contraintes syntaxiques de ces deux étapes ne

sont pas de même nature. Pour la reconnaissance de la parole, l'objectif est simplement de neutraliser des hypothèses de mots afin de restreindre la combinatoire de recherche. Pour l'interprétation l'objectif est cette fois d'aboutir à une structuration, par exemple linguistiquement motivée ou relative à des schémas pré-établis (mots clefs), permettant de déduire l'intention du locuteur.

Cette approche est la plus souvent adoptée, par exemple dans les travaux de van Noord [vN et al.98] ou au Laboratoire Central de Recherche de Thomson [Roussel et al.97]. Notons que la meilleure hypothèse acoustique fournie par le système est souvent insuffisante et qu'il est nécessaire de prendre en compte les N meilleurs hypothèses [vN et al.98] pour conserver un bon taux de reconnaissance. Afin d'illustrer l'importance de la grammaticalité suivant cette approche, une expérimentation menée à Thomson et reportée dans [Halber98a] classe les hypothèses de reconnaissance d'énoncés fournies par trois systèmes de décodage acoustico-phonétiques[15] en trois catégories :

- les hypothèses totalement incorrectes ;
- les hypothèses légérement incorrectes, c'est-à-dire des hypothèses qu'il est possible de rattrapper ;
- les hypothèses correctes.

Ces trois types d'hypothèses se répartissent de manière relativement uniforme (1/3 chacun) pour les trois systèmes de reconnaissance de la parole testés et sur un vocabulaire de 200 mots correspondant à l'application *Virtual Speaker*. L'objectif est ici d'être capable de rejeter les hypothèses totalement incorrectes, de conserver les hypothèses correctes et de rattrapper les hypothèses légérement incorrectes. Le score acoustique, c'est-à-dire le score de certitude attribué par le système de RAP, permet en pratique de discriminer les hypothèses totalement incorrectes des autres hypothèses. Il ne permet cependant pas d'identifier les deux autres catégories. C'est à ce niveau que la grammaticalité devient essentielle, car elle permet d'identifier les énoncés corrects proposés par le module de reconnaissance des énoncés légérement incorrects [Halber98a].

Intérêt d'une coopération des niveaux perceptif et interprétatif

Les analyses syntaxique et sémantique sont habituellement menées sur des séquences de mots indépendamment des détails de prononciation. Les informations prosodiques sont cependant des sources d'informations potentiellement très utiles pour détecter notamment les frontières de syntagmes, les pauses, etc. De nombreux travaux ont montré les interactions entre syntaxe et phonologie [Selkirk84].

En plus d'assigner une description de la structure d'un énoncé et une interprétation, la syntaxe et la sémantique peuvent améliorer les performances d'un système de reconnaissance de la parole. Dans toute langue, il existe des contraintes sur les séquences possibles de mots. La reconnaissance est améliorée par la qualité de modélisation de ces contraintes. L'expression de ces contraintes s'appelle *un modèle de langage*. En réduisant dynamiquement la taille du vocabulaire à prendre en compte, la combinatoire de recherche du système de reconnaissance s'en trouve réduit, et les performances améliorées. Le degré de contraintes amené par un modèle de langage se mesure de manière probabiliste sur des phrases tests par sa *perplexité*.

15. Les systèmes employés étaient capables de fournir un treillis d'hypothèses de mots pondérés par des scores de reconnaissances (scores acoustiques). Ces systèmes étaient Nuance du SRI, guidé par une grammaire, HTK d'Entropic et ABBOT des universités de Cambridge et Sheffield, tous deux guidés par un modèle de langage de type *n-gramme*.

Les travaux récents les plus performants en reconnaissance de la parole sont fondés sur des techniques stochastiques et de très important volume de données d'apprentissage afin de réduire la perplexité. Des statistiques sur les séquences de deux ou trois mots (bigramme ou trigramme) sont habituellement utilisées. Autrement dit, la probabilité qu'un mot apparaisse à un point particulier d'un énoncé est donnée sur la base d'un ou deux mots (ou classe) le précédant immédiatement (voir annexe B). Une telle estimation est intéressante pour capturer des contraintes locales, mais inefficace pour capturer des relations hiérarchiques comme les accords sujet-verbe.

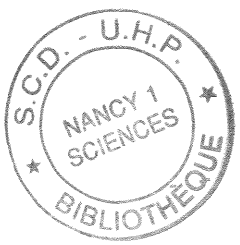
Une grammaire linguistiquement motivée permet d'exprimer des contraintes hiérarchiques et à distance. Nous verrons en particulier que l'utilisation d'une telle grammaire n'est pas incompatible avec celle de statistiques.

Il faut cependant préciser qu'une trop forte réduction de la perplexité risque de définir le langage d'application plutôt que de le modéliser. Autrement dit un modèle de langage doit absolument couvrir le sous-langage d'une application tout en maintenant une perplexité aussi basse que possible, sinon le risque est de voir le taux de reconnaissance de phrase diminuer. En outre, certains phénomènes peuvent ne pas être prévus par le concepteur d'une grammaire ou ne pas apparaître dans les corpus d'apprentissage. Ces phénomènes ne pourront alors être analysés par un système grammatical et seront rapprochés de la séquence la plus probable, mais erronée, donnée par le modèle de langage pour une approche stochastique.

Bien qu'une forte coopération entre niveaux perceptif et interprétatif puisse être mutuellement profitable, il est également difficile de mettre au point une architecture permettant de faire interagir efficacement une grammaire motivée linguistiquement et un module de reconnaissance de la parole, sachant que l'ensemble des connaissances manipulées dépend du domaine (problèmes d'ordre méthodologique). Cette approche est cependant celle menée par le laboratoire SRI depuis quelques années [Price et al.89]. Un de leurs systèmes de reconnaissance de la parole a obtenu les meilleurs résultats de reconnaissance et de compréhension lors de l'évaluation des systèmes ATIS en 1994 [Pallett et al.95], devant les systèmes de AT&T et du CMU (Carnegie Mellow University), fondés sur une intégration plus simple par treillis de mots.

Nous avons décrit les deux approches classiques d'intégration de la grammaticalité dans le processus de reconnaissance de la parole. Nous ne souhaitons pas, dans la suite de notre travail, nous imposer un de ces deux cadres de coopération. Nous avons choisi, au contraire, de nous abstraire d'un système de reconnaissance particulier et d'un processus d'intégration spécifique. Ce dernier point est plus particulièrement étudié à différent niveau de compréhension dans la thèse de David Roussel [Roussel99].

Nous avons dans cette toute dernière partie insisté sur l'importance de la grammaticalité au sein d'un système de dialogue pour la reconnaissance du langage d'application. Il est temps maintenant de s'interroger plus précisément sur ce qu'on entend par grammaticalité ou syntaxe, et sur sa mise en œuvre dans des systèmes de traitement automatique des langues.



Chapitre 3

La syntaxe

Une interrogation légitime porte tout d'abord sur la nécessité et l'importance de la syntaxe pour la compréhension de la langue. Nous tentons de répondre à ce délicat problème dans le début de ce chapitre, puis nous présentons les deux points essentiels de l'automatisation de l'analyse syntaxique d'énoncés : quelle représentation des connaissances employer et comment mettre en œuvre cette analyse ? Nous nous intéresserons ensuite aux techniques dites *robustes* d'analyse, qui nous intéressent plus particulièrement dans notre contexte de dialogue Homme-Machine, et nous justifions alors notre choix d'une approche non-déterministe et spécialisée de l'analyse grammaticale.

3.1 Rôles de la syntaxe

A la différence de l'analyse sémantique, certains chercheurs, comme Schank [Schank75], ont douté de la nécessité de passer par une étape syntaxique pour l'analyse de la langue, en privilégiant une analyse sémantique directement à partir des chaînes de mots. Nous allons tenter de montrer pourquoi l'analyse syntaxique est cependant aujourd'hui admise comme une étape indispensable en Traitement Automatique des Langues.

3.1.1 Une grammaire, deux fonctionnalités

Les travaux de Chomsky se fondent sur les deux observations suivantes [Chomsky64] :

- un locuteur s'exprimant dans une langue donnée est capable de créer une infinité de phrases ;
- ce locuteur est capable de déterminer si une phrase est « correcte » ou non et si elle a un sens.

Le but de l'analyse linguistique est d'expliquer les règles implicites qui régissent cette utilisation d'une langue, ceci invariablement de son contexte d'énonciation et de la pertinence de son utilisation. L'analyse syntaxique constitue une étape classique de l'analyse linguistique, cependant, au regard des différents courants et écoles (modèle génératif, harrissien, etc.), il est relativement difficile d'en donner une définition claire.

On peut définir de manière très générale la syntaxe comme l'ensemble des connaissances permettant d'une part de décrire les contraintes sur l'ordre des mots d'un énoncé, c'est-à-dire de donner les combinaisons autorisées et interdites, et d'autre part, d'identifier les syntagmes et les liens fonctionnels qui les unissent. On exprime le plus souvent ces connaissances sous forme de règles devant capter les régularités d'une langue donnée et

interdire les énoncés « impossibles ». L'ensemble de ces règles forme la grammaire de cette langue.

Suite à cette définition de la syntaxe, on peut mettre en avant deux aspects que nous développerons :

- la grammaire vue comme combinatoire : comment assemble-t-on les mots ?
- la grammaire présentant les sens structurels à travers l'organisation des mots dans la phrase.

Le **rôle prédictif** consiste à définir les énoncés valides et interdire les énoncés impossibles. La grammaire doit rendre compte de toutes les phrases qu'un locuteur « natif » juge grammaticales et seulement celles-la [Chomsky64] (contraintes *supra-lexicales*). Elle recouvre alors inévitablement une partie de ce qu'on appelle l'analyse morphologique, i.e. les combinaisons inférieures aux mots (contraintes *infra-lexicales*).

En psycholinguistique, on a montré depuis longtemps, [Miller et al.51], que pour un même rapport signal/bruit, les mots isolés sont moins bien reconnus que les mêmes mots dans des phrases, parce que la structure de la phrase fournit déjà des renseignements quant au type de mots qui peuvent figurer dans une position donnée. Ce qui est nécessaire chez l'homme, le sera d'autant plus pour une machine dont les capacités de perception sont loin d'égaler les performances d'une oreille humaine (de l'ordre de 70% de phonèmes reconnus pour la machine contre 90% pour l'homme).

Le **rôle descriptif** est d'identifier les syntagmes d'un énoncé et rendre compte des liens fonctionnels qui les unissent. La phrase est plus que la somme des sens de ses différents mots. Puisque d'une combinaison particulière de mots on peut déduire des fonctions particulières (qui réalise l'action, qui la subit, etc.), on recouvre ici une partie de l'analyse sémantique.

En pratique, suivant le type d'application, l'une ou l'autre de ces deux facettes sera privilégiée. Par exemple, nous l'avons vu, les modèles de langage pour la reconnaissance de la parole sur de larges vocabulaires, doivent être à même de prédire les mots possibles étant donné un préfixe correctement reconnu, ceci n'étant pas toujours possible sur la base des simples indices acoustiques. À l'inverse, pour des systèmes de traduction automatique ou d'extraction d'informations, il est impératif de pouvoir identifier les relations entre les différents constituants (par exemple que tel groupe nominal est objet direct de tel verbe).

L'analyse syntaxique est le processus qui conduit à l'identification des constituants et de leurs relations.

3.1.2 Nécessité de l'analyse syntaxique

Comme l'indique [Mela92], il est admis aujourd'hui que des systèmes de qualité de traitement automatique des langues ne peuvent faire l'économie d'une représentation sémantique des phrases et des textes, par contre la nécessité d'une approche syntaxique est parfois mise en question et certains, comme [Schank75] prônent un traitement sémantique direct des énoncés. Cependant les propriétés formelles de la langue définies jusqu'à présent ne sont pas toutes réductibles à des propriétés sémantiques. C'est ce que l'on nomme *l'autonomie de la syntaxe* et qui justifie la mise en œuvre de formalisations spécifiques à ce niveau d'analyse.

Nous avons auparavant mis en avant les deux aspects qui peuvent justifier également l'étape d'une analyse syntaxique dans une application de traitement automatique de la langue. S'en priver c'est à la fois perdre des informations sur la combinatoire des mots et une partie du sens d'un énoncé. Si l'objectif de l'application informatique n'impose

pas l'utilisation de ces informations, alors effectivement l'exploitation d'informations de nature syntaxique n'apparaîtra pas comme nécessaire. C'est le cas par exemple de systèmes basés sur une recherche d'îlots-clefs informationnels et des manipulations élémentaires de données (par exemple les systèmes ATIS [16]). Le dialogue est alors entièrement guidé par le remplissage d'un schéma sensé décrire la requête de l'utilisateur. Si on souhaite développer un système suffisamment général pour ne pas être dépendant d'une application particulière, les informations syntaxiques sont nécessaires et ne peuvent être obtenues à un autre niveau d'analyse.

Dans le cadre qui nous intéresse, c'est-à-dire celui des interfaces homme-machine utilisant la langue orale spontanée comme moyen de communication, la question de l'analyse syntaxique est également double. La composante syntaxique doit pouvoir :

- jouer un rôle prédictif sur la chaîne d'entrée : la capacité actuelle des analyseurs acoustico-phonétiques ne permet pas, en temps réel et sur un lexique correspondant à ce type d'application, d'obtenir une reconnaissance certaine. L'exploitation des contraintes syntaxiques doit pouvoir restreindre la combinatoire de recherche de l'analyseur acoustico-phonétique afin d'améliorer ses performances et apporter des hypothèses de mots reconnus ;
- jouer un rôle dans l'interprétation d'un énoncé, en mettant en évidence les syntagmes et les fonctions grammaticales associées.

Nous allons maintenant nous intéresser aux différents aspects de l'analyse syntaxique automatique. On peut considérer qu'ils reposent soit sur des problèmes de représentations des connaissances (quelle formalisation donner à la notion de grammaire ?), qui feront l'objet de la partie suivante 3.2, soit sur des questions d'algorithmique (comment mener l'analyse) étudiées dans la partie 3.3.

3.2 Modèles linguistiques

3.2.1 Les origines

Parmi les plus anciens travaux dans le domaine de la structuration linguistique des phrases, on peut citer ceux de Charles F. Hockett [17] qui a proposé d'utiliser des boîtes comme celles de la figure 3.1.

la			fillette			regardait			le			chat								
la			fillette			regardait			le			chat								
la			fillette			regardait			le			chat								
la			fillette			regardait			le			chat								
la			fille		tte	regard			ait			le			chat					

FIG. 3.1 – *Boîte de Hockett pour la représentation d'une phrase.*

16. Air Transport Information Systems. Le domaine d'application est celui de la recherche de renseignements sur les relations aériennes dans une base de données.

17. Charles F. Hockett est en fait un anthropologue étudiant la communication humaine.

Il est toutefois plus fréquent d'utiliser un parenthésage du type :

[P [SN [Dét la] [N fillette]][SV [V regardait] [SN [Det le] [Nchat]]]

Les lettres attachées aux crochets désignent les *catégories syntaxiques*, c'est-à-dire des classes permettant de regrouper des mots ayant un comportement syntaxique proche. On distingue ainsi classiquement les noms (N), les syntagmes nominaux (SN), les verbes (V) et syntagmes verbaux (SV), les adjectifs (A), etc. Mais la représentation la plus employée, équivalente à la notation par parenthèsage, est, sans aucun doute, celle d'un arbre syntaxique identifiant les différents syntagmes comme celui de la figure 3.2.

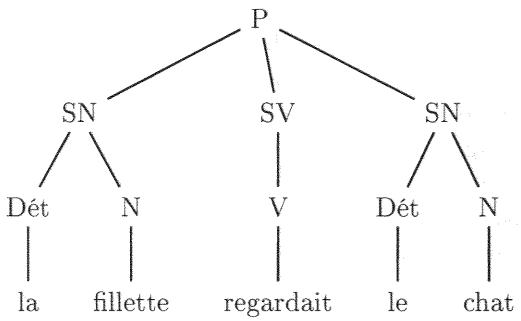


FIG. 3.2 – *Arbre syntaxique (ou syntagmatique) pour la représentation d'une phrase.*

Ces quelques représentations illustrent le principe général des modélisations linguistiques qui se sont développées tout particulièrement durant le XXème siècle : une structure formelle simple, sur nos exemples une structuration arborescente, offre un cadre général contraint dans lequel des descriptions linguistiques particulières de la phrase à décrire viennent se réaliser. Indépendamment du choix des catégories syntaxiques, qui peut dépendre de la langue, on pose comme principe la possibilité de décrire tous les constituants d'une phrase et toutes les dépendances relatives de ses constituants [Tesniere59] sous une forme arborescente.

Dès 1957, Chomsky, au lieu d'une description des constituants par la décomposition de chaque phrase d'un corpus, propose des règles universelles permettant de générer un infinité de phrases à partir d'un nombre fini d'éléments, comme pour l'exemple suivant :

Ph ->(SP) SN SV (SP)
SN ->(Dét) N (SP) (SA)
SN ->(SN) (SP)
SV ->(AUX) V (SN) (SP) (SA)
SP ->Prép SN
SA ->(SAdv) A (SP)
SAdv ->(SAdv) Adv
N ->Pro

De modèles descriptifs, on passe alors à des modèles ayant pour ambition d'une part d'expliquer le processus de production des phrases (grammaire générative [Chomsky64] et grammaire de chaîne de Harris [Harris51]) et, d'autre part, de capter les contraintes liant

les différents syntagmes. S'appuyant sur une théorie mathématique pour représenter les règles linguistiques régissant la composition des constituants, réduisant au maximum la part d'implicite, il devenait également envisageable de se baser sur ces formalisations pour développer des traitements automatiques de la langue (analyse et génération).

L'annexe A rappelle quelques notions de théorie des langages associées aux travaux de Chomsky, ainsi que les premiers formalismes de représentation linguistiques ayant abouti à des réalisations informatiques.

3.2.2 Les formalismes actuels importants

La formalisation des représentations des connaissances linguistiques a connu de très importants progrès au début des années 80. Les grands traits de cette évolution portent à la fois sur le plan formel et sur les descriptions linguistiques. On peut noter :

- la **lexicalisation** d'un formalisme syntaxique consiste à associer à chaque entrée lexicale une modélisation des contextes syntaxiques dans lesquels elle peut être utilisée. Lexique et grammaire se confondent alors en un lexique syntaxique. L'intérêt de la lexicalisation est double : tout d'abord, la possibilité d'associer à chaque entrée lexicale un degré de finesse des descriptions particulièrement souple, évitant les effets de bords de grammaire fondée sur une régularité de la langue (illusoire dès que l'on traite de la langue naturelle même finalisée, comme l'ont montré les travaux du LADL). Ensuite, cette propriété permet d'appliquer des heuristiques d'analyse, renvoyant de nombreux problèmes d'ambiguïté à des ambiguïtés lexicales plus simples à traiter.
- l'**unification** a permis, par son expressivité, d'améliorer le contrôle d'un nombre important de phénomènes linguistiques comme les mécanismes d'accords ou de sous-catégorisation [Shieber86]. Habituellement, l'utilisation de l'unification consiste à ajouter des contraintes sur un squelette de règles hors contextes (on parle alors de grammaires syntagmatiques[18]) à l'aide de DAG[19] (Graphes orientés acycliques). C'est le cas pour les formalismes des Grammaires Lexicales Fonctionnelles (LFG[20]), des Grammaires Syntagmatiques Généralisées (GPSG[21]) et des Grammaires Syntagmatiques guidées par les Têtes (HPSG[22]). Suivant le même principe, le formalisme des Grammaires Lexicalisées d'Arbres Adjoints (LTAG[23]) associe des structures de traits à un squelette plus puissant qu'une grammaire hors contexte exprimée à l'aide d'arbres partiels d'analyse plutôt que par des règles de réécriture.

Un type de grammaire est caractérisé par les capacités génératives *faible* et *forte*. La capacité générative *faible* d'une grammaire est son pouvoir de discrimination entre phrases acceptées et phrases rejetées, autrement dit elle définit l'ensemble des phrases qui constitue le langage décrit [Gardent et al.93]. La capacité générative *forte* est son pouvoir de construction d'une représentation, c'est à dire sa capacité à associer à chaque phrase une structure plus ou moins complexe.

18. *Phrase structure grammars*
19. *Directed Acyclic Graphs*
20. *Lexical Fonctionnal Grammars [Kaplan et al.82]*
21. *Generalized Phrase Structure Grammars [Gazdar et al.85]*
22. *Head-driven Phrase Structure Grammars [Pollard et al.94]*
23. *Lexicalized Tree Adjoining Grammars [Joshi et al.75]*

3.2.3 Approches guidées par la logique

Initié par les premiers travaux de Colmerauer pour le traitement des langues, ce type de formalisation correspond à une approche ancienne qui a conduit notamment à la conception du langage de programmation *Prolog*. Les Grammaires à Clauses Définies (DCG[24]) ont connu et connaissent encore un certain succès. L'analyse est ici vue comme une déduction logique à partir d'axiomes et de termes qui sont les structures d'information de base des DCG.

On peut aussi citer des approches basées sur la logique linéaire, mais encore à l'état de recherche (il n'existe pas de système linguistique à large couverture fondé sur ce formalisme). Les approches guidées par la logique présentent de plus des limites de robustesse face aux énoncés oraux, notamment parce que le contrôle de l'analyse est difficile.

3.2.4 Les grammaires catégorielles

Le modèle des grammaires catégorielles a été proposé à la suite des travaux de Husserl par les logiciens polonais S. Lesniewski et Z. Adjukiewicz au début des années 30, pour l'analyse sémantique et syntaxique des langues. Dans ce modèle, les unités linguistiques se voient assignées un type (sémantique ou syntaxique). Chaque type est représenté sous forme d'une « fraction formelle », la vérification de la bonne formation des expressions linguistiques revient alors à appliquer une procédure de résolution analogue à la simplification des fractions dans les nombres rationnels. Le logicien H.B.Curry en 1949 a appliqué ce modèle à l'analyse syntaxique des langues. J. Lambek a, par la suite, étendu le modèle dans un calcul sur les types des expressions linguistiques (calcul L des types). Il a en particulier introduit des règles de composition des types et des règles de changement de types (i.e. de catégories).

Pendant plusieurs années, les Grammaires Catégorielles n'ont reçu aucun développement important. D'autres modèles, tout particulièrement les modèles transformationnels de Chomsky et Harris, ont capté l'attention des linguistes. Depuis le début des années 80, les grammaires catégorielles connaissent un regain d'intérêt, principalement à la suite des travaux de R. Montague, et se sont enrichies des nouvelles possibilités offertes par le principe d'unification. Le modèle UCG[25] par exemple a été conçu en vue d'exploiter tout particulièrement ce dernier point et la lexicalisation. Les propriétés formelles et linguistiques du calcul de Lambek ont pu être explorées pour la description de constructions comme la coordination, que les grammaires catégorielles classiques (comme le modèle AB) étaient incapables d'analyser. Plusieurs grammaires catégorielles sont actuellement expérimentées en syntaxe et en morphologie, mais tout en étant également loin du degré d'achèvement et d'efficacité informatique des formalismes LFG, HPSG ou LTAG.

3.2.5 Limites des modèles linguistiques

Il n'existe actuellement aucune grammaire couvrant la totalité des phénomènes syntaxiques d'une langue. Certains phénomènes fréquents restent mal connus et mal décrits. Nous pouvons citer en particulier :

- *la coordination* : cette relation n'est, par définition, pas hiérarchique et ne peut être captée simplement par une structure arborescente ;

24. *Definite Clause Grammars [Pereira et al.80]*
25. Unification Categorial Grammars [Baschung91].

- *l'ellipse* : la fréquence de ce phénomène est particulièrement importante, nous l'avons vu, dans le cadre de dialogues oraux, or les formalismes grammaticaux existants ignorent bien souvent ce phénomène[26] ;
- *les circonstanciels* et les problèmes de double portée : là encore la limite vient d'une représentation hiérarchique. Par exemple dans la phrase *j'ai entendu chanter ce matin*, le circonstanciel *ce matin* porte à la fois sur *entendre* et *chanter* ;
- *les verbes modaux* qui posent une difficulté dans le choix de la hiérarchisation : les modaux doivent-ils être considérés comme modifieur ou comme tête syntaxique dans le syntagme verbal où ils apparaîssent ?
- *les expressions figées et semi-figées* où les interprétations compositionnelle et non-compositionnelle sont concurrentes.

D'autre part, l'objectif même de ces formalismes, décrire les constructions langagières à l'aide d'un petit nombre d'opérateurs et des unités homogènes bien définies, est vite mis à défaut dans des applications informatiques à large couverture. Les constructions non canoniques sont très fréquentes dans des textes tout-venant ou la langue parlée. Si on souhaite vraiment décrire tous les cas de figure, on génére vite énormément de solutions parasites sur les phrases les plus banales, puisque toutes les règles, celles fréquentes et celles qui le sont moins, sont mises au même plan [Fuchs et al.93].

3.2.6 Intérêt des modèles linguistiques

Les modèles linguistiques présentent un nombre important d'intérêts pour des applications informatiques : tout d'abord ils permettent d'identifier les constituants, leurs dépendances mutuelles et leurs fonctions. De plus, ces modèles permettent une large couverture de la langue, des solutions pour l'intégration des niveaux supérieurs de traitements (immédiat avec HPSG par exemple), une finesse dans les descriptions initiales et dans les résultats obtenus, et, le plus souvent, d'importantes ressources linguistiques associées. Ce dernier point semble être une des conditions nécessaires pour la pérénnité des travaux sur un formalisme. Même pour des applications dont le domaine est restreint, se fonder sur un formalisme linguistique, qui se veut général, permet de porter plus facilement un système vers une autre application.

Le choix du formalisme est essentiel pour la représentation et la compréhension des phénomènes linguistiques. Cependant d'autres approches, dites guidées par l'informatique, au vue des domaines d'application circonscris qu'elles visent, privilégient avant tout la robustesse des résultats sur la modélisation linguistique. Étant donné nos objectifs, on peut légitimement se demander s'il nous faudrait plutôt privilégier l'adéquation linguistique du modèle proposé ou sa robustesse. La section suivante, consacrée aux enjeux de l'*analyse* syntaxique, c'est-à-dire la mise en œuvre des connaissances du modèle linguistique, va nous éclairer plus précisément sur ce compromis entre adéquation linguistique et robustesse des procédés d'analyse. Nous verrons ensuite dans quelle mesure il est possible de satisfaire ces deux impératifs.

26. Cette observation est d'autant plus sensible lorsqu'on sait que bien des phénomènes d'ellipses sont réduits en anglais à des anaphores, et que les formalismes grammaticaux dominants ont tous été conçus par des anglo-saxons.

3.3 Problèmes et enjeux de l'analyse syntaxique automatique

3.3.1 Techniques d'analyse grammaticale

La grammaire est ici la spécification statique d'un langage (au sens mathématique elle définit l'ensemble des phrases de ce langage). Rien n'est dit quant à la façon dont une phrase est susceptible d'être analysée. Cette dernière tâche est assurée par un analyseur qui associera à une séquence de mots la ou les structures syntaxiques correspondantes. On utilise parfois aussi la notion de *reconnaisseur* dont la tâche, plus simple, est de décider si oui ou non une phrase est reconnue par la grammaire. L'utilité d'un reconnaisseur pour le traitement de la langue naturelle est pour le moins limitée, mais sa réalisation peut être vue comme un sous-but en vue de la conception d'un analyseur[27].

Un tel système qui effectue une distinction entre grammaire et analyseur est appellé *système déclaratif*. La grammaire est une donnée externe qui sera interprétée ou compilée. Elle peut ainsi être modifiée sans remettre en cause l'implantation d'un analyseur. D'autres systèmes, par exemple les ATN[28] (Réseaux de Transitions Augmentés), ne font pas cette séparation et incorporent simultanément les données grammaticales et leur utilisation. Ils sont appellés systèmes *procéduraux*. L'analyseur est conçu pour une grammaire particulière, chaque règle faisant l'objet d'une procédure. Une modification de la grammaire entraînera une modification de l'analyseur.

L'évolution des techniques mises en œuvre durant les vingt dernières années est tout à fait significative : si les analyseurs procéduraux étaient courants durant les années 70, par exemple dans les systèmes de dialogue, citons Myrtille 2 [Pierrel81] ou HWIM [Wolf et al.80], les recherches se concentrent maintenant sur le développement de systèmes déclaratifs. Les raisons sont de trois ordres :

- théorique : les systèmes procéduraux ne respectent pas la distinction introduite par Chomsky entre compétence et performance, la *compétence* recouvrant la connaissance qu'un individu possède de sa langue, tandis que la *performance* concerne l'utilisation de ce savoir ;
- logistique : les systèmes procéduraux sont difficiles à développer et à maintenir, ils sont rarement portables et peu modulaires ;
- algorithmique : dû à son caractère procédural, à l'ordre strict des mots que le formalisme impose, des techniques d'analyse efficace et robuste ne peuvent pas être employées. [Carroll83] montre en effet que l'analyse automatique des dépendances est très complexe et qu'une stratégie d'analyse autre que strictement de gauche à droite est inutilisable pour une grammaires ATN à large couverture.

Les systèmes déclaratifs ont pour origine les grammaires hors contextes de Chomsky, qui peuvent être vue comme l'extrême opposé des ATN. Des techniques d'analyse pour les règles de réécriture introduites par Chomsky existent, elles sont employées, nous l'avons évoqué, dans les compilateurs de langage informatique. Cependant le formalisme, insuffisamment contraint, est d'une expressivité beaucoup trop limitée [Shieber86] et ses utilisations pratiques pour l'analyse de la langue naturelle sont très rares. C'est plus récemment, avec l'émergence de nouvelles théorie déclaratives, de nature beaucoup plus contrainte, comme LFG ou HPSG, que l'approche procédurale a décliné. Bien que linguistiquement

27. On peut noter qu'obtenir un analyseur à partir d'un reconnaisseur n'est pas toujours immédiat et n'est parfois pas possible sans dégradation de la complexité d'analyse.

28. Augmented Transitions Networks [Woods70]

plus élégantes et théoriquement mieux fondées, ces théories présentent des difficultés pratiques pour l'analyse automatique lorsqu'il s'agit de mettre en œuvre des grammaires à large couverture de la langue.

Il n'est ainsi pas étonnant, en parallèle à cette évolution, d'assiter à de très importants progrès en matière de techniques algorithmiques pour l'analyse syntaxique grammaticale. De nombreux problèmes comme la récursivité infinie et le traitement des ambiguïtés ont trouvé des solutions élégantes avec l'utilisation d'un *chart*, sur lequel nous allons revenir en 3.3.4.

3.3.2 La question des ambiguïtés

L'analyse syntaxique est très vite confrontée à plusieurs possibilités d'analyse qui peuvent correspondre :

- à des situations d'ambiguïtés globales effectives (exemples classiques : *le joueur de football américain*, *la belle ferme le voile*, *le boucher sale la tranche* ou encore *la petite brise la glace*) ;
- à des situations d'ambiguïtés artificielles, liées au formalisme ou à la stratégie d'analyse, on parle de *surgénération*.

À l'origine de ces ambiguïtés, nous pouvons distinguer différents problèmes :

- le problème de la segmentation en phrases d'un énoncé oral : à quelle phrase rattacher, par exemple, un complément circonstanciel situé entre deux phrases ? Des éléments prosodiques peuvent apporter des informations sur ce point ;
- le problème du découpage de syntagme, par exemple, dans l'énoncé *l'exposé de soutenance de thèse de Laurent*, *de Laurent* se rattache-t'il à *exposé*, *soutenance* ou *thèse* ?
- le problème de l'identification de fonction : il est parfois difficile de déterminer quelle fonction associer à un syntagme. Par exemple, en cas d'absence de préposition : *il chante le matin*, ou pour déterminer le sujet d'infinitive : *il promet à Jean de venir* (le sujet de l'infinitive est *il*), et *il permet à Jean de venir* (le sujet de l'infinitive est *Jean*). Dans ce dernier exemple, l'ambiguïté est artificielle, un analyseur devra disposer de nombreuses informations propres aux verbes *promettre* et *permettre* pour éviter des constructions syntaxiques incorrectes. Dans cet autre exemple : *j'ai vu souvent manger des poulets*, les poulets sont-ils mangeurs ou mangés ?

Comme le rappelle [Carroll93], deux des problèmes majeurs de l'analyse syntaxique automatique sont, tout d'abord, le fait que l'analyse d'énoncés réalistes avec des formalismes syntaxiques sophistiqués comme ceux que nous avons présentés dans la partie 3.2, avec l'énumération de toutes les ambiguïtés reste très couteuse en temps de calcul. D'autre part, si de nombreuses analyses sont assignées à un énoncé, peu sont sémantiquement acceptables et, en contexte, seulement une analyse constitue la base d'une interprétation correcte.

3.3.3 Le difficile compromis couverture/robustesse/efficacité

L'objectif de l'analyse syntaxique automatique d'un énoncé est de lui associer une ou plusieurs *dérivations* décrivant sa structure, c'est-à-dire identifiant ses différents constituants et les rôles que ces derniers entretiennent entre eux. Étant donné une grammaire et un énoncé, une dérivation est un chemin parmi l'ensemble des constructions pouvant être

généré par cette grammaire, d'une configuration initiale à une configuration finale dans laquelle tous les constituants élémentaires ont été consommés. Ces constituants élémentaires correspondent habituellement aux mots, bien que la notion de mot reste informelle, tout particulièrement pour des langues à morphologie complexe. Un énoncé peut se voir assigner plusieurs dérivations, reflétant autant d'ambiguïtés syntaxiques. En plus du compromis, lié à l'application, entre prédictivité qui fait appel à une grammaire très contrainte sur les suites de mots acceptables et assignation d'une ou plusieurs dérivations pour un maximum d'énoncés, d'autres caractéristiques sont propres au processus de recherche des dérivations.

Plus le nombre de règles est important, plus la *couverture* de l'analyseur est importante, c'est-à-dire plus la proportion de phrases effectivement utilisées dans une langue ayant une dérivation dans la grammaire est grande.

La *robustesse* d'un analyseur est sa capacité à fournir des dérivations partielles dans le cas d'un énoncé hors de la grammaire. L'idée est d'être capable d'en extraire les sous parties grammaticales.

L'*efficacité* de l'analyseur est sa capacité à donner cette ou ces dérivations le plus rapidement possibles, et donc repose sur des notions de complexité d'algorithme. En pratique, ces derniers dépendent de la longueur de la chaîne d'entrée à analyser (habituellement notée n) et sur la taille de la grammaire utilisée (notée G). Plus le nombre d'éléments lexicaux et de règles mis en jeu est important, plus la couverture augmente, plus l'efficacité est compromise par l'augmentation du facteur de complexité G. Une solution est de diriger l'analyse par le but afin de restreindre l'espace de recherche. L'inconvénient porte alors sur la robustesse puisque les cas agrammaticaux ne sont même pas envisagés.

3.3.4 Déterminisme de l'analyse

L'analyse syntaxique, nous l'avons vu, peut s'assimiler à une recherche des dérivations qu'il est possible d'associer à un énoncé étant donné une grammaire. La taille de l'espace de recherche est un facteur essentiel pour l'efficacité d'une application informatique et dépend du formalisme syntaxique considéré. Cette taille dépend de deux facteurs : la structure de l'espace de recherche induite par la grammaire et l'exhaustivité de la procédure de recherche.

Sur ces deux points, deux problèmes importants sont à prendre en compte par l'analyseur : éviter tout d'abord les solutions parasites, et d'autre part la réitération de calculs identiques. Plusieurs réponses sont envisageables, l'analyseur peut renvoyer la première solution trouvée (moyennant quelques choix pertinents), ou encore l'ensemble des cas d'ambiguïtés effectives, qui pourront être soumis, éventuellement en parallèle, à l'analyseur sémantique qui déterminera laquelle est sémantiquement correcte.

Une langue naturelle étant intrinséquement ambiguë, il est normal que la grammaire sensée la décrire soit ambiguë, et donc difficile qu'un analyseur déterministe puisse être envisagé à moins d'intégrer directement des informations contextuelles. Cependant, parmi ces ambiguïtés, il est essentiel de pouvoir écarter les cas d'ambiguïtés artificielles locales et les solutions parasites.

Méthodes déterministes

Les deux méthodes les plus classiques mises en œuvre pour gérer ces problèmes sont celle du retour en arrière et du parallélisme. Le *retour en arrière* (*back-tracking*) consiste à choisir l'une des solutions (en général selon certaines heuristiques) et à continuer l'analyse

en mémorisant les solutions non retenues et l'état d'analyse au moment du choix. Si ce choix s'avère mauvais et mène à une impasse, on reprend l'analyse avec une des solutions mémorisées (ce qui correspond à un *back-tracking* dans l'espace de recherche). Le temps de traitement de cette méthode peut être important et dépend pour beaucoup des heuristiques mises en œuvre au moment des choix. De plus, cette stratégie ne permet pas d'obtenir l'ensemble des ambiguïtés effectives puisqu'elle s'arrête, en principe, à la première solution, et peut effectuer plusieurs fois des calculs identiques.

Plutôt qu'un retour en arrière, les méthodes employées mettent généralement en œuvre une technique prédictive, c'est-à-dire un « regard en avant » sur la présence d'éléments qui suivent l'élément sur lequel porte l'ambiguïté. Il s'agit par exemple des algorithmes déterministes *LR(k)*, où *k* donne le nombre de mots sur lequel porte ce regard en avant. D'autres analyseurs *déterministes*, comme celui proposé par [Marcus80] ou le système *Fidditch* [Hindle89], utilisent des fonctions d'évaluation sophistiquées pour déterminer si oui ou non il faut calculer l'expansion d'une hypothèse en cours. Cette approche reste très empirique : elle ne peut pas assurer qu'aucune hypothèse correcte ne sera éliminée. Un analyseur de ce type pour l'analyse du français à l'aide d'une grammaire à large couverture de la langue a également été développé, il s'agit du système ANDI du LIMSI [Sabah89].

Analyse par *chart*

La seconde méthode est celle du *parallèlisme*, elle consiste à développer l'ensemble des solutions possibles lorsqu'un choix se présente. Ici le retour en arrière est superflu dans la mesure où la dérivation correcte est toujours présente dans l'ensemble des alternatives possibles. Tous les cas d'ambiguïtés effectives sont alors déterminés et on évite les réitérations de calcul identique, mais cette stratégie présente l'inconvénient d'entraîner des temps de calcul importants en l'absence d'un véritable traitement parallèle puisqu'alors le parallélisme est simulé et qu'il faut parcourir l'ensemble de l'espace de recherche.

C'est dans cette approche que s'inscrit l'analyse par *chart* que nous allons maintenant présenter plus en détail. Les grammaires employées présentent un degré de localité important dans la mesure où les transitions d'une configuration à une autre ne font pas intervenir l'ensemble de la configuration de départ, mais seulement certaines parties bien délimitées. Dans ce cas les dérivations peuvent être factorisées en sous-dérivations indépendantes, et tabulées, c'est-à-dire indexées dans une table appellé *chart*, en vue d'une réutilisation. La construction des dérivations partageant les mêmes sous-dérivations en est grandement accélérée. De tels algorithmes tabulaires sont très largement utilisés en analyse et pour les modèles de langage avec des grammaires appropriées, citons pour les grammaire hors contextes, les algorithmes ascendants [Younger67], [Kay86], prédictifs [Earley70] [Lang74], et LR [Tomita87]. Ces algorithmes permettent une recherche exhaustive d'un nombre potentiellement exponentiel de dérivations tout en assurant une complexité au pire des cas polynomiale en temps et en espace par rapport à la longueur de la chaîne à analyser. De plus, les algorithmes tabulaires peuvent être étendus à l'aide de techniques de programmation dynamique pour une recherche optimale des dérivations en considérant une fonction d'évaluation des analyses.

La tabulation est immédiate pour les grammaires à états finis, les entrées de la table correspondant à un état et une position d'entrée. Les grammaires hors contextes sont un exemple standard dont les dérivations peuvent être tabulées. D'autres classes de grammaires, comme les grammaires légèrement dépendantes du contexte [Joshi et al.91] et certaines grammaires contraintes (HPSG par exemple) sont également adaptées à un pro-

cessus d'analyse tabulaire [Shieber92]. D'une manière générale, si analyse grammaticale il y a, l'utilisation de techniques tabulaires, qui sont maintenant bien connues, est inévitable si on souhaite la mise en œuvre d'une analyse efficace et exhaustive de l'espace de recherche.

3.3.5 Stratégie et contrôle de l'analyse

L'analyse syntaxique dans le domaine informatique a été particulièrement développée pour la compilation des langages de programmation et c'est dans ce cadre que de nombreuses méthodes furent mises au point. Un certain nombre de règles de réécriture constitue la grammaire du langage.

L'analyse *descendante* (dite *dirigée par les hypothèses*) consiste à aller des buts aux faits, c'est-à-dire aux mots. Partant de l'axiome P, on applique successivement une règle de réécriture sur l'élément le plus à gauche et on compare à chaque étape le résultat avec la phrase à analyser. Les avantages de cette méthode sont les suivants :

- elle ne génère pas de structures syntaxiques qui se révèleront impossibles à rattacher par la suite (réduction dynamique de l'espace de recherche) ;
- elle évite des ambiguïtés liées au choix entre plusieurs catégories : seules les catégories compatibles avec une analyse complète seront mises en jeu.

Cependant, sans « regard en avant », cette stratégie est susceptible d'effectuer de mauvais rattachements d'ajouts (par exemple de relatives) et, de plus, les résultats partiels obtenus posent d'importants problèmes de robustesse : les énoncés agrammaticaux stoppent l'analyse sans qu'elle ait pu en extraire le maximum d'informations possibles.

L'analyse *ascendante* (dite *dirigée par les données*), consiste à partir de chaque mot, à chercher à le remplacer par une des catégories qui peuvent lui être associées, puis à parcourir les règles de réécriture du membre droit vers le membre gauche pour construire la structure syntaxique jusqu'à remonter à l'axiome P. L'avantage de cette méthode est sa flexibilité en cas d'erreur. Cependant, en générant de nombreuses hypothèses, elle tend à occuper plus de place en mémoire et nécessite plus de traitements. Par exemple il faut remplacer les mots par leur catégorie, si un mot appartient à plusieurs catégories, comme « *le* » qui est à la fois article et pronom, on est devant autant de possibilités à analyser. De plus, pour la même raison, cette stratégie est susceptible de causer des ambiguïtés artificielles.

Analyses ascendante et descendante se heurtent à des difficultés de natures différentes, c'est pourquoi un certain nombre de statégies mixtes ont été développées afin de palier aux inconvénients de ces deux méthodes. Un autre facteur pouvant influencer l'analyse est le sens de lecture : analyse droite-gauche, gauche-droite ou bidirectionnelle. Il est également possible de procéder sans suivre l'ordre dans lequel les mots se présentent. La technique des *îlots de confiance* est souvent mise en œuvre par les analyseurs ascendants : l'analyseur cherche alors à analyser les segments non ambigus, quelles que soient leurs positions dans la phrase.

3.4 Approches robustes

Les techniques classiques d'analyse présentées dans la section précédente se révèlent peu efficaces dans le cas de phrases présentant des structures inattendues comme les énoncés oraux. La raison en est bien sûr que, durant des années, les recherches en TALN se sont basées sur des analyses complètes d'énoncés grammaticaux. D'autre techniques, on parle parfois de formalismes guidés par l'informatique, sont apparues au cours de ces dernières années, privilégiant la robustesse sur les fondements théoriques linguistiques.

3.4.1 Analyse par segments

Cette approche de l'analyse syntaxique emploie des grammaires représentées sous la forme de cascade d'automates ou de transducteurs à états finis. Les expressions régulières sont habituellement écrites à la main. Ces systèmes ne produisent en général qu'une seule sortie en employant une heuristique simple fondée sur le choix du segment le plus étendu pour résoudre les cas d'ambiguïtés lorsque plus d'une expression régulière couvre la chaîne d'entrée à une position donnée.

Les deux principaux analyseurs mettant en œuvre ces techniques sont :

- l'analyseur déterministe *Fidditch* [Hindle83] qui a été utilisé pour la conception du projet de corpus annoté syntaxiquement de l'Université de Pennsylvanie (*Penn Treebank*) ;
- CASS [Abney90], fondé sur les principes d'analyse par segments grammaticaux qui ont été évoqués dans la partie 1.3.2. Il s'agit d'un analyseur multi-niveau mettant en œuvre des processus simples et peu coûteux à chaque étape pour parvenir à une analyse. Les trois principales étapes mises en œuvre sont les suivantes : une identification des groupe nominaux et verbaux (obtenue avec un taux de réussite de l'ordre de 95%), puis des rattachements prédicats-arguments avec des processus de réparation éventuels pour les ellipses par exemple, et enfin les rattachements de modifieurs (adjectifs, compléments circonstenciels, adverbes, etc.).

Les résultats fournis par ce style d'analyseurs sont intéressants dans la mesure où ils se prêtent particulièrement bien à des processus de réparation (simple combinaison d'îlots) pouvant être guidés par la sémantique. De plus, ces résultats ne sont pas contraints par une représentation arborescente qui s'avère vite trop rigide pour représenter des analyses d'énoncés en langue spontanée.

3.4.2 Les méthodes sélectives et leurs limites

Les grandes réussites de ces approches, assez largement employées, concernent avant tout des domaines où une analyse complète n'est pas nécessaire : extraction d'information, terminologie, etc. De très nombreux travaux ont étendu ces techniques aux systèmes de dialogue homme-machine avec une réussite incontestable, par exemple sur le français au LIMSI [Lamel98] et à l'IRIT [Brison et al.95].

Ces méthodes peuvent être vues comme l'application directe des analyses par segments pour les systèmes de dialogue homme-machine. Elles correspondent aux systèmes à mots clefs (caractéristiques des systèmes ATIS) et à la version améliorée dite des *segments conceptuels* [Boros et al.96]. Ces approches se limitent à l'extraction de mots clefs ou de prédicats simples et ignorent le reste de l'énoncé. L'idée est par exemple de repérer la

présence d'une préposition spatiale, puis d'appliquer sur la suite de l'énoncé une grammaire locale de lieux dépendante de l'application et représentée à l'aide d'une expression régulière.

Cette démarche est limitée à des domaines d'application restreints, les dialogues informatifs pour lesquels la phase de compréhension se limite le plus souvent à un simple remplissage de schémas en vue de faire avancer la tâche. Dans d'autres domaines applicatifs, comme les dialogues de commande, l'interprétation des énoncés est plus complexe avec en particulier la question de la résolution des référents et du rattachement propositionnel, qui peut nécessiter une véritable analyse syntaxique. Même pour des systèmes informatifs de réservation de voyage, Daniel Luzzati relève que quelquefois les lieux d'arrivée et de départ sont intervertis [Luzzati89]. Dans le cas de dialogues mulimodaux, cette ambiguïté est encore plus sensible avec l'omniprésence des prépositions spatiales. Enfin, cette solution reste très dépendante du domaine et du type d'application réalisé. Pour [Antoine et al.99], les méthodes de ce type ne semblent donc pas être en mesure de franchir à court terme la barrière d'une analyse détaillée du langage.

3.4.3 Analyse robuste probabiliste

Le point commun de ces approches est de se fonder sur une modélisation stochastique pour parvenir à une meilleure robustesse d'analyse. Trois types d'approches se dégagent des travaux existants :

- les *grammaires stochastiques*, où les probabilités sont employées afin de désambiguïser l'analyse en favorisant les structures les plus probables. Le principe théorique de ces grammaires est présenté en annexe B. Dans ce style de systèmes, on part habituellement de grammaires linguistiquement très lâches et très générales, dont le pouvoir prédictif est pris en charge par les probabilités. Autrement dit, la surgénération d'hypothèses qui serait induite par une telle grammaire est évitée par un filtrage des dérivations les moins probables [Lari et al.90] ;
- les travaux d'extension des modèles de langage, de type N-gramme, en vue d'identifier les fonctions syntaxiques et les dépendances syntaxiques. On parle souvent pour ces approches d'analyses superficielles stochastiques (*stochastic shallow parsing*). *Shallow parsing* est un terme générique pour les analyses qui sont moins complètes que les analyseurs conventionels fondés sur un formalisme linguistique. La sortie de tels analyseurs *superficiels* n'est pas une structure d'arbre syntagmatique, mais une identification de constituants, comme les groupes nominaux, sans indication de leur structure interne, et avec éventuellement leurs rôles fonctionnels dans l'énoncé [Jelinek et al.94] ;
- les modèles purement inférés statistiquement comme DOP (Data Oriented Parsing) [Bod95], dans ce cas des constructions syntaxiques, sous la forme de fragments d'arbres, sont déduits de fréquences d'apparition dans un corpus. Les estimations de fréquence associées aux fragments d'arbres sont utilisées pour ordonner les étapes de dérivation et retrouver ainsi l'arbre d'analyse le plus probable.

Ces différentes techniques ont pu être expérimentées grâce à la disponibilité de vastes corpus annotés syntaxiquement (*treebanks*) en langue anglaise (plusieurs millions de mots). Il faut cependant noter que la langue française ne dispose toujours pas de tels corpus et que l'application de ces méthodes pour un analyseur syntaxique sur le français n'est pas

encore possible[29].

On peut aussi relever, suivant ces méthodes auto-adaptatives, les approches neuro-mimétiques (par exemple le système MICRO [Antoine96]). Notons enfin que des paramètres statistiques sont également utilisés en complément des méthodes de sélection conceptuelle présentées dans la section précédente.

3.4.4 Amélioration de l'analyse par chart

Cette approche se fonde sur des combinaisons d'îlots correctement reconnues en relachant certaines contraintes d'analyse, par exemple sur les ordres de mots ou leurs positions. On peut voir ces techniques comme un compromis entre qualité des résultats (ambiguïté) basée en général sur des principes linguistiques, efficacité des techniques tabulaires et robustesse (relâchement des contraintes locales pour faire progresser l'analyse).

Des algorithmes tabulaires particuliers ont été développés afin de réaliser la recherche préliminaire d'îlots. Citons l'algorithme *Head Corner* (analyse guidée par les têtes), où, durant une première phase ascendante, pour chaque règle de réécriture est choisie une catégorie qui portera la prédiction d'un îlot. Habituellement cette catégorie est celle qui correspond à la tête sémantique du syntagme. L'algorithme *Left Corner* (guidé par le coin gauche) est une cas particulier de l'algorithme *Head Corner* où la catégorie tête est systématiquement la première rencontrée à gauche. Ces deux algorithmes s'avèrent plus efficaces qu'un algorithme purement ascendant comme CKY et susceptible de fournir des résultats comparables, c'est-à-dire un ensemble d'îlots correctement reconnus. Un algorithme *Head Corner* a été employé dans le système OVIS[30] [vN et al.98] avec d'excellents résultats en temps d'analyse et en couverture.

Une approche complémentaire est de réaliser une analyse par *chart* plus tolérante, c'est le cas de l'algorithme GLR* proposé par [Lavie96], suivi de l'application, dans une seconde passe, de règles de réparation spécifiques [Rose et al.97]. Cette approche est particulièrement souple car elle permet de combiner les avantages d'une analyse grammaticale, généralement optimisée à l'aide de probabilités et de processus de réparation plus *ad hoc*.

Parmi les réalisations notables ayant adopté cette approche, citons également l'analyseur par *chart* étendu de Seneff [Seneff92], qui permet le recouvrement de groupes analysés dans une phrase complète. Cet analyseur est utilisé avec le système de reconnaissance de la parole du MIT du domaine ATIS. Le système PLNLP [Jensen et al.93] repose sur une architecture incrémentale, toujours par *chart*, intégrant syntaxe et sémantique et destiné à l'analyse et la génération de langue naturelle sur de larges vocabulaires. Ces réalisations montrent qu'en plus de la généralité qu'elle permet d'un domaine à l'autre et de la finesse des résultats qu'il est possible de fournir, cette approche est tout à fait compétitive et viable pour des applications automatiques soumises à des contraintes de robustesse et de temps de réponse.

Nous allons maintenant tenter de justifier et caractériser plus précisemment comment mettre en œuvre cette approche pour l'analyse d'énoncés oraux.

29. La francophonie est sur ce point très en retard par rapport à ses voisins, notons cependant qu'un tel corpus est actuellement en cours de développement à TALaNa [Abeille et al.99].

30. OVIS : *Openbaar Veroer Informatie Systeem*

3.5 Pour une approche non-déterministe et spécialisée de l'analyse grammaticale

3.5.1 Modèles grammaticaux et modèles stochastique inférés

Les travaux de Van Noord sur le projet OVIS (système de renseignements sur les transports publics) [vN et al.98] ont démontré l'efficacité d'une analyse grammaticale fine de treillis de mots pour des systèmes de dialogue, par rapport aux méthodes plus superficielles fondées sur la détection de concepts et les approches purement stochastiques. En particulier, sur ce projet, son approche était en concurrence avec celle, prometteuse, de Rens Bod de l'Université d'Amsterdam [Bod95], fondée sur l'inférence automatique de grammaire d'arbres probabilistes (modèle DOP). Nous rapportons dans la table 3.1 la comparaison des résultats obtenus suivant ces deux approches pour un millier de treillis de mots. La méthode *DOP 1* autorise des fragments d'arbres de profondeur maximale 2, la méthode *DOP 2* de profondeur maximale à 4. La grammaire employée est une grammaire HPSG simplifiée [vN et al.98], analysée à l'aide d'un algorithme de type *Head Corner* [vN97]. La méthode *grammaticale 1* emploie un bigramme pour identifier le meilleur chemin sur le graphe de mots, la méthode *grammaticale 2* un trigramme. Le premier résultat donne le pourcentage d'énoncés entièrement analysés, le second le pourcentage d'énoncés dont on a identifié la structure sémantique correcte. Enfin, les deux dernières colonnes indiquent le temps total d'analyse (en secondes) et la taille mémoire des méthodes employées (en Megabytes) (voir [vZ et al.99] pour plus de détails).

Méthode	% énoncés analysés	% sémantique	temps (s)	taille mémoire (Mb)
DOP 1	69,3	74,9	7011	619
DOP 2	69,4	74,9	32798	621
grammaticale 1	75,9	81,3	340	38
grammaticale 2	76,5	82,1	1723	64

TAB. 3.1 – *Comparaison des approches grammaticales et des méthodes DOP dans le système OVIS [vZ et al.99].*

L'approche grammaticale s'y distingue très largement. En plus de distancer les performances du modèle DOP, elle offre des résultats supérieurs à la plupart des méthodes sélectives sur des domaines similaires (dont elles sont dépendantes).

3.5.2 Intérêts du non déterminisme des niveaux de traitement

Pour l'analyse robuste de la parole, nous l'avons vu, plusieurs stratégies d'analyse ont été proposées, dont le but est de *reconnaître* une analyse unique par des méthodes sélectives [Boros et al.96], stochastiques [Schwartz et al.96] ou heuristiques [Boufaden98]. Par analogie avec [Marcus80], on note que les éléments suivants sont caractéristiques de ces approches :

- l'analyse est prévue pour délivrer un seul résultat ;
- toutes les structures retenues en fin d'analyse correspondent aux éléments d'une même analyse ;
- aucune structure temporaire n'est générée comme résultat de l'analyse.

Dans le cadre de l'analyse robuste pour les systèmes de dialogue parlé, ce type de déterminisme est requis par des contraintes de performance, et les capacités de traitement limitées

(volontairement ou de fait) des modules d'interprétation.

A l'évidence, même si ce choix est peu suivi en pratique, une stratégie de type non déterministe conviendrait mieux à un système de dialogue qui dispose d'une interface vocale. Plusieurs arguments sont en faveur d'un point de vue non déterministe, et donc d'une analyse potentiellement exhaustive de l'espace de recherche :

- la tâche de décodage de la séquence de parole ne disposant pas de connaissances suffisantes pour garantir le résultat, l'analyseur ne doit pas non plus délivrer une analyse unique correspondant à une hypothèse unique de reconnaissance. Il semble au contraire préférable de ne pas écarter trop tôt les différentes analyses possibles car le coût d'une erreur de compréhension est élevé : elle engage des interactions verbales supplémentaires pour lesquelles des erreurs potentielles sont à nouveau à envisager.
- une stratégie qui sélectionne, parmi les N meilleures hypothèses délivrées par le système de reconnaissance, la première qui maximise un critère de bonne formation donné n'est pas satisfaisante. Ce critère possède un faible pouvoir discriminant sur les hypothèses de mots concurrentes. Sur les hypothèses de phrase, le succès de cette stratégie est aussi variable : il est tributaire d'une hypothèse de rareté des séquences reconnues qui sont conformes au critère de bonne formation considéré. Si erreur de reconnaissance il y a, c'est bien parce qu'il existe une incompatibilité entre l'énoncé et les modèles acoustiques (défaut de prononciation, bruit) ou entre l'énoncé et le modèle de langage qui guide la reconnaissance de la parole (mot hors vocabulaire, construction mal représentée dans le corpus qui a servi à l'apprentissage du modèle de langage, etc.). Il n'est donc pas toujours pertinent de se fier au rang des hypothèses délivrées par le système de reconnaissance.

3.5.3 Analyse d'un usage de la langue plutôt que de la langue générale

Dans l'état de l'art sur la syntaxe proposé dans [Abeille et al.97], les auteurs insistent sur la nécessité, afin d'améliorer les analyses, de prendre en compte les différents usages de la langue dans la mesure où aucun système ne pourra traiter une langue naturelle dans sa totalité. Dans le cadre de la communication homme-machine, nous l'avons vu, il est nécessaire de tenir compte de plus de spontanéité et de la restriction au domaine. Nous considérons qu'une telle remarque doit pouvoir s'accompagner de la définition d'une méthodologie pour spécialiser une grammaire pouvant être générale à une application et un domaine particulier, ceci afin de rendre l'adaptation de l'analyseur vers un autre domaine aussi économique que possible.

3.5.4 Techniques de *déterminisation* de l'analyse grammaticale

Afin d'éviter la forte combinatoire associée à l'exploration complète de l'espace de recherche, l'analyse s'appuie sur des heuristiques, qui peuvent être d'ordre psycholinguistiques, comme par exemple l'analyseur à pile de [Shieber83] (on parle alors *d'analyseurs cognitifs*), ou d'ordre stochastique, les probabilités donnant des principes préférenciels sur un sous-langage.

Les méthodes stochastiques constituent un apport important pour la prise en compte des variations. Leur intégration dans un algorithme d'analyse syntaxique tabulaire permet d'exploiter des connaissances générales de la langue et du domaine sous la forme de contraintes de préférences, contrairement à d'autres modélisations stochastiques fondées sur des n-grammes où toute combinaison d'étiquettes ou segments conceptuels, par

exemple, est rendue possible.

Cependant, bien que l'application de ces optimisations sur un analyseur nous semble importante et doit s'intégrer à notre approche, elle ne semble, pour le français, pas encore possible étant donné l'absence de corpus étiquetés syntaxiquement.

3.5.5 Difficultés d'implantation

Un problème important concerne la généralité et la réutilisabilité des processus d'analyse [Abeille et al.97]. Les implantations se doivent d'une part d'être aussi proches que possible du modèle théorique, par essence général, quitte à spécialiser par la suite son utilisation pour un type de langue particulier. D'autre part, l'utilisation et la durée de vie d'une implantation dépendent de sa généricité, de sa portabilité, et plus généralement des choix techniques mis en œuvre. Nous considérons même qu'il vaut mieux pécher en trop de généralisation qu'en une réalisation *ad hoc*.

3.6 Bilan

3.6.1 Un modèle procédural de l'analyse syntaxique du français parlé

Au final nous avons pu cerner l'importance de l'analyse syntaxique, mais il n'y pas de formalismes propres au traitement de l'oral (lié aux manques de modèles de l'oral et, certainement, à la nature même d'un formalisme linguistique) ni de techniques vraiment satifaisantes. Les applications devant traiter de la langue orale spontanée se basent sur des méthodes dites robustes, principalement fondées sur des probabilités (signe de notre incapacité à modèliser les phénomènes) et sur des méthodes sélectives destinées à des dialogues simples de type informatifs et dépendantes du domaine.

Nous avons, dans les chapitres précédents, soutenu l'idée de traiter l'oral à partir de techniques de l'écrit. Du point de vue du traitement automatique des énoncés oraux, l'intérêt principal d'une telle approche est la possibilité d'utiliser les ressources et les techniques informatiques initialement conçues pour le traitement de l'écrit et de les adapter pour l'oral. De plus, nous pouvons nous appuyer sur les principaux travaux empiriques menés en linguistique sur le français parlé, [BB90]. Nous souhaitons proposer un modèle procédural de l'analyse syntaxique de la langue orale, inspiré par ces études. Nous considérons d'abord que nous disposons d'une grammaire rassemblant les connaissances statiques générales de la langue. Une analyse syntaxique à partir de cette grammaire correspond, nous l'avons vu, à un développement suivant l'axe syntagmatique, aux phénomènes elliptiques et aux effacement près. Une première interrogation porte sur la place de ces deux phénomènes dans un système complet. Les ellipses et omissions se manifestant par un manque dans les structures syntaxiques d'une grammaire non-elliptique, il semble plus adapté de représenter ces phénomènes directement dans la grammaire en considérant certaines contraintes argumentales comme optionnelles, et en identifiant directement les éléments susceptibles de donner lieu à un effacement (ce phénomène ayant été observé ou non sur corpus). Suivant ce principe, les segments grammaticaux les plus étendus localiseront les variations paradigmatiques. Au vue des solutions existantes, le choix le plus naturel et le plus sûr est de rétablir l'axe syntagmatique avec des règles de réparation fonctionnant sur des îlots correctement reconnus à l'aide d'un algorithme tabulaire. Une fois l'axe syntagmatique restauré par les règles de réparation, l'analyse grammaticale pourra reprendre l'extension

des îlots jusqu'à la localisation d'une autre variation syntagmatique ou jusqu'à l'analyse complète de l'énoncé.

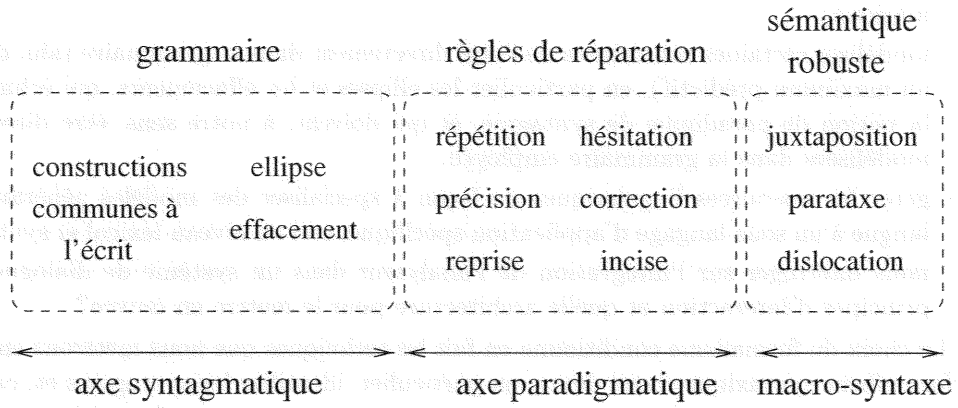


FIG. 3.3 – *Approche générale du traitement syntaxique de l'oral.*

La figure 3.3 illustre les phénomènes et les traitements correspondants considérés. Nous reprenons les trois axes de développement de [BB90], axe syntagmatique, axe paradigmatique et axe correspondant à la macro-syntaxe. Au premier axe de développement, nous l'avons vu, correspond une analyse grammaticale délivrant les segments grammaticaux les plus étendus. Étant donné le non-déterminisme de cette analyse, le choix d'une analyse par *chart* permettra de la mener efficacement. La grammaire employée ici sera une grammaire construite selon des principes de l'écrit à laquelle nous intégrerons des contraintes liées aux possibilités d'ellipses et d'effacements. Le second axe sera traité par des règles de réparation ramenant les réalisations orales des syntagmes à des réalisations grammaticales. Un troisième niveau a pour charge de traiter la macro-syntaxe, c'est-à-dire de procéder aux bons rattachements argumentaux dans le cas de phénomènes de fragmentation de l'énoncé.

Nous considérons que les phénomènes d'incises tels que définis dans le premier chapitre de ce mémoire, relèvent d'un traitement paradigmatique. Ceci est contestable puisque qu'une incise ne correspond pas à un paradigme de syntagme et n'a qu'un rôle fonctionnel dans le dialogue. Cependant nous avons considéré pour la suite qu'elles relevaient du processus de production de l'oral, et n'appartenant pas à l'axe syntagmatique, ne pouvaient être intégrées à la grammaire. De plus, comme pour les variations paradigmatiques, elles correspondent à un arrêt de l'axe de développement grammatical, et seront donc identifiées, comme les variations paradigmatiques, par l'adjacence de deux segments grammaticaux non rattachables.

3.6.2 Objectifs

Compte tenu de l'approche choisie, des techniques et formalismes existants pour l'écrit, nous pouvons définir nos objectifs, et tenter de combler ce qui manque aux techniques plus habituellement appliquées à l'écrit, de la façon suivante :

- analyser les parties grammaticales des énoncés en s'appuyant sur un formalisme évolué dédié à l'écrit et sur des techniques tabulaires robustes d'analyse ;
- appliquer des techniques de factorisation et des heuristiques pour diminuer l'espace de recherche ;

- mettre en œuvre des traitements adaptés aux distorsions de l'oral pour couvrir certaines agrammaticalités. Ceci correspondra au traitement des variations paradigmatiques selon [BB90] et aura pour conséquence de rétablir correctement l'axe syntagmatique ;
- modéliser certaines contraintes de l'oral directement dans la grammaire (afin de rester au maximum prédictif), en particulier les ellipses et les effacements, qui échappent à la notion de paradigme de syntagme, et qui doivent, à notre sens, être directement modélisées dans la grammaire employée ;
- gérer les ressources linguistiques de façon à spécialiser des modèles généraux de la langue à un sous-langage d'application spécifique, ceci au niveau lexical et syntaxique ;
- nous interroger sur l'intégration de l'analyseur dans un système de dialogue : quels principes d'interaction et quelle architecture pour le mettre en œuvre ?

Le choix du formalisme conditionne en fait les techniques que nous mettrons en œuvre. Le formalisme syntaxique choisi devra, en particulier, identifier les syntagmes et, condition que nous avons exprimée au cours de cette partie, permettre une analyse délivrant les segments grammaticaux les plus étendus. Le formalisme devra également rendre possible l'intégration de techniques de réparation, afin de traiter les variations de l'axe paradigmatique. Enfin, la mise en œuvre d'une analyse globale, équivalente au niveau macro-syntaxique de [BB90], capable de donner les différentes dépendances des constituants de l'axe syntagmatique, faisant appel à des connaissances de niveaux différents de la syntaxe, devra être possible à partir des résultats fournis par l'analyseur robuste. Nous allons donc présenter et justifier dans la partie suivante le choix du formalisme adopté qui est celui des Grammaires Lexicalisées d'Arbres Adjoints.

Deuxième partie

Les Grammaires Lexicalisées d'Arbres Adjoints

Chapitre 1

Présentation du formalisme LTAG

Nous présentons et justifions dans ce chapitre le modèle linguistique choisi, celui des Grammaires Lexicalisées d'Arbres Adjoints (LTAG). Nous avons vu dans le chapitre précédent qu'un formalisme syntaxique pouvait se caractériser par un système formel et par les principes linguistiques se réalisant sur ce système. Dans le cas des LTAG, le modèle formel fut introduit bien avant que des principes linguistiques précis lui soient associés. C'est déjà là une des particularités attrayantes de ce modèle. Avec LFG ou GPSG, ce sont avant tout des principes linguistiques qui ont trouvé une réalisation dans un appareil formel dédié (synchronisation d'un squelette structurel hors contexte et de contraintes plus puissantes par DAG). Ici, c'est au contraire à partir de la base mathématique du formalisme LTAG que de nombreuses études sur l'expressivité linguistique du modèle ont été menées. En pratique, ceci a amené une richesse de points de vue et d'approches à la fois en linguistique et en informatique dites *fondées sur les TAG.*

Le formalisme TAG est donc un formalisme mathématique, mais, comme pour les grammaires hors contexte, il a été conçu en vue de modéliser la langue et suivant des intuitions linguistiques. Pour cette présentation, nous tenterons de justifier les grandes caractéristiques et choix qui ont conduits à la définition du formalisme, sans respecter l'ordre chronologique de l'apparition des différents concepts[31].

1.1 Une description formelle des LTAG

1.1.1 Le mécanisme d'analyse

Le formalisme des Grammaires d'Arbres Adjoints a été défini tout d'abord dans [Joshi et al.75], sous le nom initial de *Tree Adjunct Grammar.* L'objectif d'Aravind Joshi, ancien élève du fondateur des grammaires de chaînes, Z. Harris, était d'étendre les opérations d'extension de chaînes, alors caractérisées par des règles d'écriture hors contexte, par des opérations d'extension d'arbres. Il s'agissait alors de gagner en expressivité structurelle, ainsi que de localiser et caractériser plus précisemment les dérivations. En effet, les grammaires hors contexte ont connu leur succès avec les langages formels de programmation, grâce à l'efficacité des traitements qu'elles permettent et à la simplicité des automates qui leur correspondent. Mais, elles se sont révélées d'une expressivité trop limitée dans le domaine du traitement de la langue.

31. Pour cette présentation nous nous sommes inspirés en particulier des tutoriaux du workshop TAG+4, Philadelphie, 1998, notamment ceux d'A. Joshi et O. Rambow, ainsi que de la présentation de [Abeille93]

Dans le formalisme des LTAG, une dérivation, c'est-à-dire une composition d'unités syntaxiques, ne se caractérise plus comme une chaîne obtenue par d'autres chaînes mais comme un arbre obtenu d'autres arbres.

Dans le cas d'une grammaire arborescente, les structures élémentaires sont des arbres partiels d'analyse. Ces arbres partiels se combinent en vue d'obtenir un arbre des composants (ou arbre d'analyse) classiques. En effet une grammaire hors contexte peut être vue comme un système générant des arbres, chaque règle de réécriture caractérisant un seul niveau père/fils de l'arbre comme le montre la figure 1.1 :

$$\begin{aligned} SN &\rightarrow \text{Jean} \\ P &\rightarrow SN\ V \\ V &\rightarrow \text{mange} \end{aligned}$$

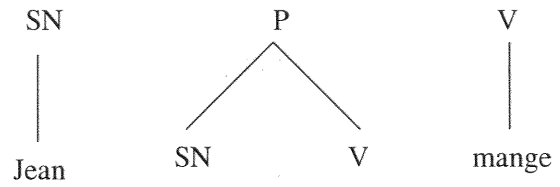


FIG. 1.1 – *Exemple d'équivalence entre grammaire hors contexte et grammaire arborescente de niveau de profondeur 1.*

L'élément de base d'une grammaire arborescente est un arbre partiel d'analyse appelé *arbre élémentaire*, dont la profondeur peut être quelconque.

Notons bien que nous obtenons alors un système générant des arbres à partir de chaînes mais ce système reste aussi peu contraint qu'une simple grammaire hors contexte. Des principes supplémentaires doivent être associés à cette grammaire arborescente.

Si on autorise une profondeur quelconque pour les arbres élémentaires, une grammaire arborescente présente une première propriété spécifique et très intéressante appellée *domaine de localité étendu*. Cette propriété correspond à la possibilité de définir des contraintes dans un arbre partiel d'analyse à plusieurs niveaux d'arborescence, donc au-delà d'une simple règle de réécriture, mais toujours de manière structurelle. Ainsi des contraintes non-locales pour une grammaire hors contexte pourront trouver une réalisation locale dans un arbre élémentaire. Notons que d'autres formalismes comme LFG tentent aussi de rendre locales des contraintes échappant à une grammaire hors contexte mais en faisant appel à des mécanismes potentiellement exponentiels comme des DAG.

Lexicalisation

Nous avons vu dans la partie précédente 3.2.2 l'importance donnée au lexique au sein des formalismes syntaxiques modernes. Que ce soit pour LFG ou HPSG, les informations spécifiquement lexicales viennent directement influencer la syntaxe. L'intégration du lexique et de la syntaxe est discutée et justifiée dans [Abeille91b] ou dans les travaux du

LADL[32] [Gross75].

La lexicalisation pour une grammaire arborescente impose que chaque arbre élémentaire contienne au moins un nœud terminal particulier appellé *ancre* ou *tête* occupé par un mot. Un arbre élémentaire peut contenir plusieurs ancres, on parle alors de *co-ancres* ou de *co-têtes*, sachant que l'une d'elle est désignée comme ancre principale.

Ce principe permet d'associer à chaque entrée lexicale l'ensemble des structures qui caractériseront ses emplois possibles. Lexique et grammaire se confondent alors en un lexique syntaxique. Associée au principe du domaine de localité étendu, cette propriété est particulièrement intéressante puisqu'elle permet une description précise des structures et contraintes syntaxiques en fonction des mots qui les ancrent. La lexicalisation est intéressante également d'un point de vue informatique, puisqu'étant donné une phrase, on peut se limiter pour son analyse au sous-ensemble des arbres élémentaires de la grammaire effectivement ancrés par les mots de la phrase. Nous verrons cependant que la lexicalisation n'est pas sans contrepartie.

Afin d'exploiter cette propriété de lexicalisation, le problème est maintenant de savoir quelles sont les règles de composition d'arbres c'est-à-dire les opérations de dérivation qui permettront la lexicalisation complète d'une grammaire arborescente, tout en maintenant un pouvoir génératif satisfaisant. L'opération de substitution offre un premier élément de réponse.

La substitution

La substitution correspond à l'opération de réécriture pour une grammaire hors contexte. Elle permet d'insérer un arbre initial ou dérivé d'un arbre initial, à un nœud de substitution d'un arbre élémentaire ou dérivé qui est noté par $\downarrow$. La substitution à un nœud terminal de substitution est toujours obligatoire. Elle est illustrée figure 1.2.

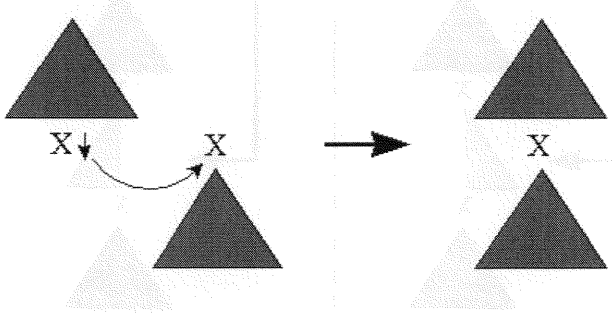


FIG. 1.2 – *Opération de substitution*

Si on s'en tient à cette simple opération de combinaison d'arbres et à la propriété de lexicalisation, nous obtenons une *Grammaire de Substitution d'Arbres* (TSG). Ce type de grammaire présente cependant deux grandes insuffisances qui le rend difficilement exploitables :

- limite formelle : une grammaire hors contexte ne peut pas être lexicalisée en une TSG[33] ;

32. Laboratoire d'Automatique Documentaire et Linguistique, Paris.

33. Notons ici que la forme normale de Greinbach ne donne qu'une lexicalisation faible d'une grammaire CFG,

- limite linguistique : même en considérant une grammaire hors contexte pouvant être lexicalisée par une TSG, les arbres associés aux modifieurs rendent difficiles toute exploitation linguistique.

Ceci justifie l'introduction d'une seconde opération de composition d'arbre afin de permettre la lexicalisation de la grammaire : l'adjonction.

L'adjonction

L'adjonction permet d'insérer un arbre présentant certaines caractéristiques à un nœud interne ou racine d'un arbre élémentaire ou dérivé. L'adjonction entre arbres au niveau d'un nœud obéit à des contraintes catégorielles et peut être interdite, obligatoire ou, il s'agit du cas le plus fréquent, facultative et réitérable. On distingue alors dans une grammaire LTAG deux types d'arbre : les arbres pouvant s'insérer dans d'autres arbres appellés *arbre auxiliaire* et ceux ne le pouvant pas appellés *arbre initial*. Les arbres auxiliaires répondent à la contrainte particulière suivante : ils contiennent un unique nœud spécial appellé *nœud pied* de même catégorie que le nœud racine de l'arbre auxiliaire. Le nœud auquel vient s'insérer l'arbre auxiliaire est appellé nœud site de l'adjonction et ne peut être un nœud feuille.

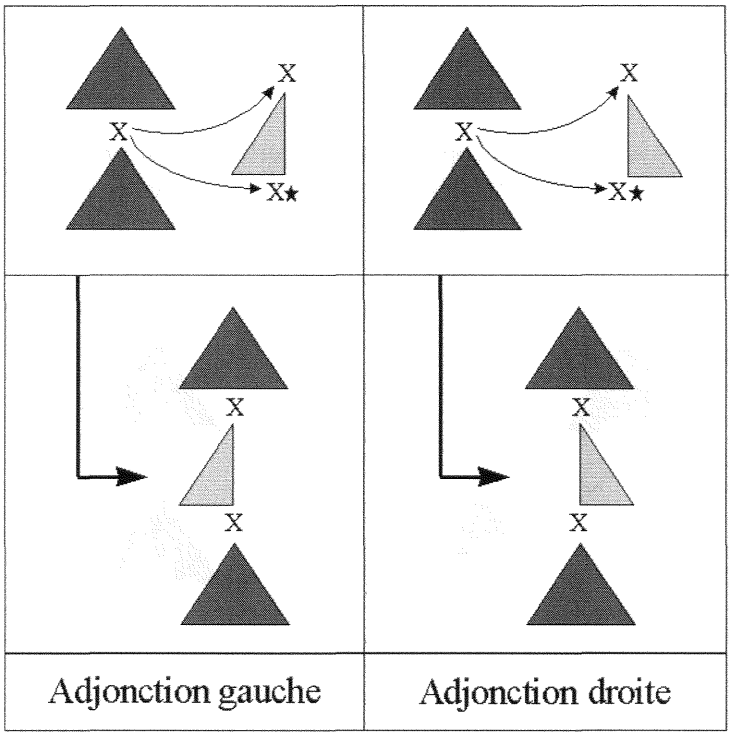


FIG. 1.3 – *Opérations d'adjonctions (ou insertions) gauche et droite.*

On appelle *épine dorsale* d'un arbre auxiliaire (*spine*) le chemin reliant le nœud pied au nœud racine de l'arbre. Si aucun nœud ne se trouve à droite de l'épine dorsale d'un

c'est-à-dire que l'ensemble des chaînes donné par la grammaire obtenue est identique à la grammaire initiale, mais pas l'ensemble des descriptions structurelles (autrement dit pas l'ensemble des arbres).

arbre auxiliaire, on parle alors d'arbre auxiliaire *gauche*, réciproquement d'arbre auxiliaire *droit* s'il n'y a aucun nœud du coté droit de l'arbre auxiliaire. Dans [Schabes et al.95b] une opération d'adjonction où l'arbre auxiliaire est droit (resp. gauche) est appellé adjonction ou *insertion* droite (resp. gauche). Ces deux opérations sont représentées sur la figure 1.3. En particulier si on limite une grammaire d'arbre à la présence d'arbres auxiliaires gauche ou droit, et si on interdit la construction au cours de l'analyse d'arbre auxiliaire autre que droit ou gauche, on obtient une Grammaire d'Arbres Insérés (TIG) lexicalisée équivalente à une grammaire hors contexte lexicalisée [Schabes et al.95b] en termes de capacité générative forte et faible. L'adjonction permet donc de lexicaliser une grammaire hors contexte.

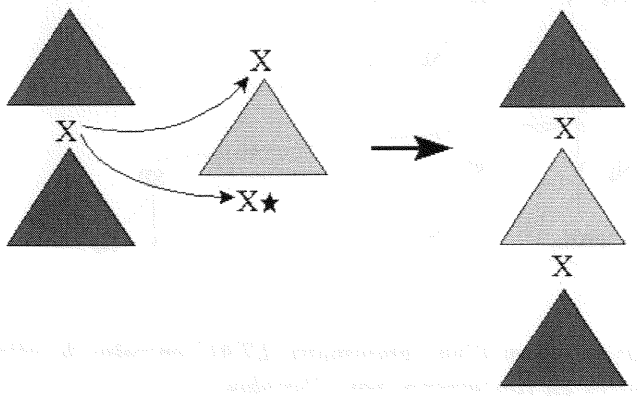


FIG. 1.4 – *Opération générale d'adjonction*

Lorsqu'un arbre auxiliaire propose du matériel à la fois à droite et à gauche du nœud pied, on parle d'arbre auxiliaire *englobant* et d'adjonction *englobante* (*wrapping adjunction*). Cette opération est alors plus puissante que la substitution ou les adjonctions gauches ou droites et est spécifique au formalisme LTAG. La capacité générative du formalisme dépasse alors celle d'une grammaire hors contexte, et est caractéristique d'une famille de formalisme intermédiaire entre grammaire hors contexte et dépendante du contexte appellée grammaire légérement dépendant du contexte. D'autres formalismes comme les LIG (Linear Indexed Grammars) ou les CDG (Constraint Dependency Grammars) appartiennent également à cette classe. Le mécanisme général d'adjonction est illustré figure 1.4.

Deux définitions différentes ont été proposées pour caractériser les dérivations obtenues par adjonction, celle de [VS87] et la définition alternative de [Schabes94]. La justification de ces choix repose en fait sur ce qu'on attend des structures résultantes que nous allons maintenant présenter.

1.1.2 Résultats d'une analyse

L'analyse d'une phrase à l'aide d'une grammaire TAG donne tout d'abord comme résultat des *arbres dérivés*, qui fournissent les structures syntagmatiques associées à la phrase obtenue par combinaison des arbres élémentaires par adjonction et substitution. Notons que chaque mot peut ancrer plusieurs arbres, comme le montre la figure 1.5, et

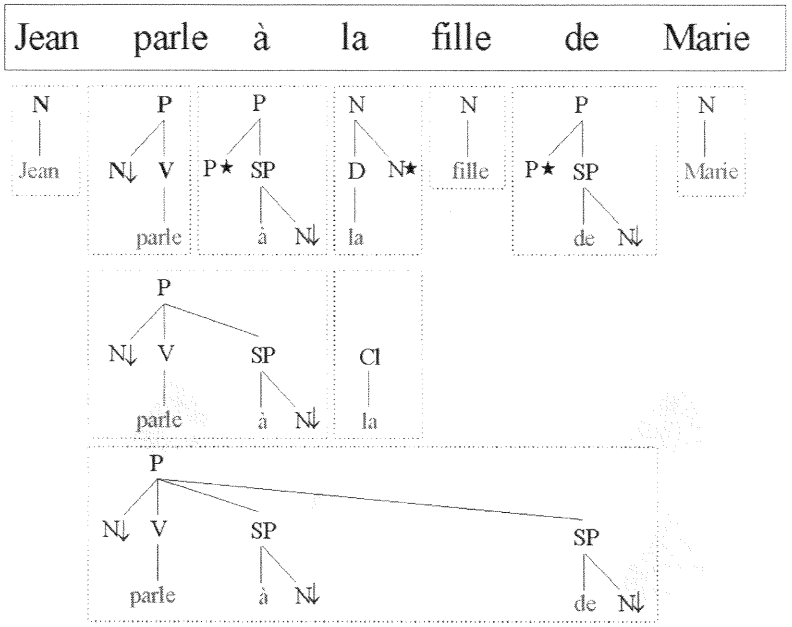


FIG. 1.5 – *La lexicalisation d'une grammaire LTAG entraîne la sélection d'un ensemble d'arbres élémentaires en concurrence pour l'analyse.*

c'est à la charge de l'analyse syntaxique d'étudier l'ensemble des combinaisons possibles afin d'aboutir à ces résultats. La figure 1.6 donne deux exemples d'arbres dérivés pour une même phrase à analyser.

Pour les grammaires utilisant des règles de réécriture hors contextes, la structure syntagmatique d'une phrase indique sans ambiguïté à partir de quelles règles elle a été obtenue. A l'inverse, pour les grammaires arborescentes, un arbre dérivé ne permet pas forcément de déduire de manière unique à partir de quels arbres élémentaires cette structure a été obtenue. L'analyse fournit ainsi un second type de résultat, les *arbres de dérivation* dont les nœuds sont étiquetés par des arbres élémentaires et qui indiquent par quelles combinaisons d'arbres élémentaires chaque arbre dérivé a été obtenu. Dans l'arbre de dérivation que nous présenterons, on note par un trait pointillé l'adjonction et par un trait plein la substitution (voir figure 1.7). Alors que l'arbre dérivé permet de saisir les constituants composant un énoncé, l'arbre de dérivation permet de capter les dépendances entre ces constituants.

Un arbre de dérivation est dit *complet* si tous les nœuds feuilles de l'arbre dérivé correspondant sont des ancres. Dans la définition initiale des dérivations de [VS87], un même nœud ne peut recevoir qu'au plus une adjonction. Cette contrainte permet de déduire d'un arbre de dérivation un unique arbre dérivé. L'inconvénient de cette définition est qu'en associant des principes de correspondance entre arbres élémentaires et prédicats sémantiques, l'arbre de dérivation ne correspond alors pas tout à fait à un arbre de dépendance. Notons que c'est cette définition qui est couramment utilisée dans les implantations du formalisme.

La définition proposée par [Schabes94] permet plusieurs adjonctions à un même nœud et, pour obtenir toujours un seul arbre dérivé par arbre de dérivation, impose d'ordonner

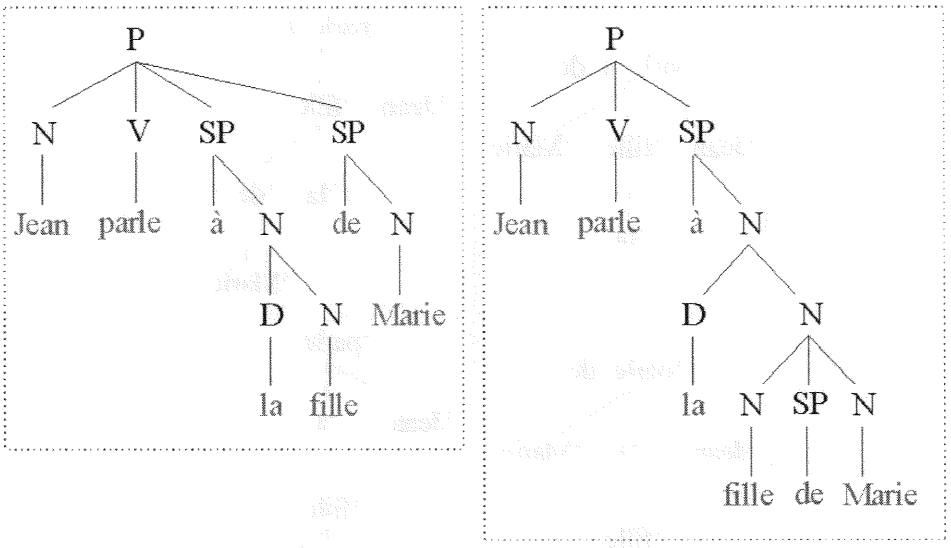


FIG. 1.6 – *Exemples d'arbres dérivé*

les branches de l'arbre de dérivation. Nous verrons que cette définition se prête mieux à l'interprétation sémantique dans la mesure où l'arbre de dérivation peut être vu comme un arbre des dépendances sémantiques.

Quelle que soit la définition sur laquelle ils reposent, les arbres de dérivation constituent une représentation proche de la structure argumentale de la phrase analysée (qui correspond à la structure fonctionnelle dans les LFG), et donc représentent un bon squelette d'analyse sémantique prédicative. C'est donc bien l'arbre de dérivation qui constitue le résultat significatif d'une analyse, l'arbre dérivé n'étant en somme qu'une structure de travail.

1.1.3 Mécanismes d'unification

Afin d'exprimer plus facilement des contraintes sur les rattachements possibles d'arbres, l'usage des structures de traits a été défini pour les grammaires TAG dans [VS87] et [VS et al.88]. La propriété de localité étendue permet de se limiter à l'utilisation de traits atomiques, c'est-à-dire de couple *(attribut, valeur)*, *valeur* pouvant être une valeur du domaine de l'attribut ou une variable.

A chaque nœud interne et racine d'un arbre élémentaire ou dérivé est associé deux groupes de traits :

- les traits *top* ou traits *d'amont*, qui indiquent les relations du nœud avec les noeuds qui le dominent ;
- les traits *bottom* ou traits *d'aval*, qui indiquent les relations du nœud avec les noeuds qu'il domine.

Habituellement une seule structure de traits est utilisée au nœud n'accueillant pas d'adjonction c'est-à-dire les nœuds de substitution et les nœuds pieds, pour ces derniers toutefois on associe tout de même également deux ensembles. Ces répartitions des traits en deux structures pour chaque nœud interne et racine sont liées à la possibilité de réaliser une adjonction.

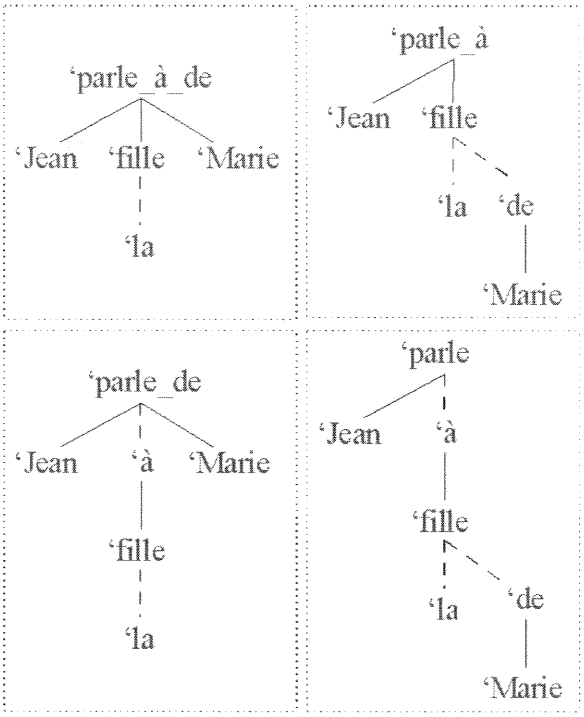


FIG. 1.7 – *Exemples d'arbres de dérivation*

A la fin d'une opération de dérivation, les structures de traits *top* et *bottom* sont mis à jour de la façon suivante :

- en cas de substitution : les traits top du nœud racine de l'arbre substitué doivent s'unifier avec les traits du nœud où il y a eu substitution, voir figure 1.8 ;
- en cas d'adjonction : on doit avoir, d'une part, unification des traits top du nœud racine de l'arbre auxiliaire avec les traits top du nœud recevant l'adjonction, et, d'autre part, unification des traits du nœud pied de l'arbre auxiliaire avec les traits bottom du nœud recevant l'adjonction, voir figure 1.9.

A la fin d'une analyse, pour chaque dérivation complète obtenue, parties *top* et *bottom* doivent s'unifier à chaque nœud de l'arbre dérivé correspondant.

Notons ici une propriété importante pour l'analyse : la valeur finale des variables de

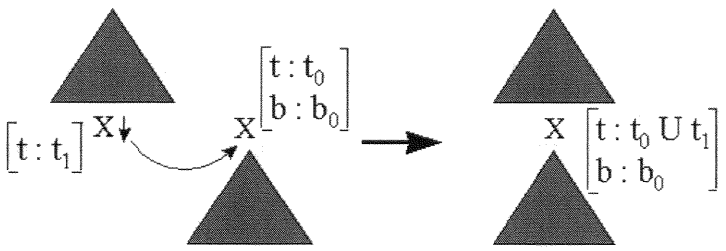


FIG. 1.8 – *Unification pour l'opération de substitution.*

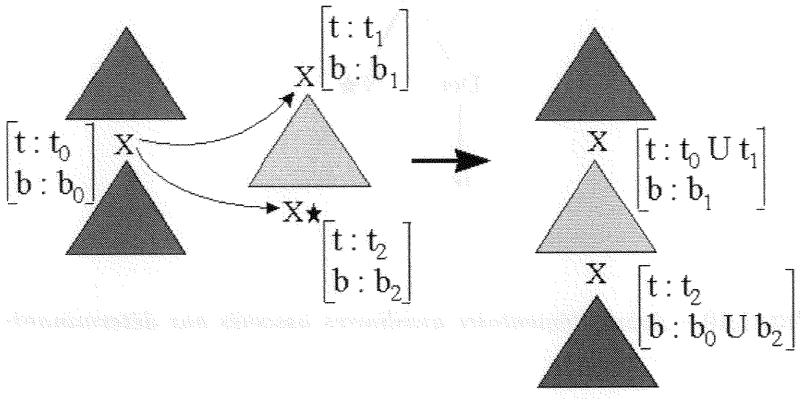


FIG. 1.9 – *Unification pour l'opération d'adjonction.*

ces structures de traits dépend de la façon dont ont été combinés les arbres élémentaires. Le nombre de valeurs finales possible des traits d'un arbre dérivé est égal au nombre de dérivations possibles. L'analyse de LTAG augmentée de structures de traits est donc au pire des cas exponentielle dans la mesure où le nombre maximal de dérivations pour un formalisme lexicalisé comme les LTAG est exponentiel par rapport à la longueur de la chaîne à analyser.

1.1.4 Contraintes d'adjonction

Afin de contrôler de façon plus souple les adjonctions, on associe souvent aux nœuds potentiellement sites certaines contraintes, qui peuvent être de trois types :

- les contraintes de non-adjonction (notés *na*) interdisent toute adjonction sur les nœuds ainsi contraints ;
- les contraintes sélectives d'adjonction n'autorisent une adjonction que pour un sous-ensemble des arbres auxiliaires de la grammaire ;
- enfin les contraintes d'adjonction obligatoires sont habituellement réalisées avec les structures de traits.

Ainsi, par exemple, les déterminants, qui sont traités à l'aide d'une adjonction sur le nom, porteront sur le nœud racine N une marque de non-adjonction, caractérisant le fait qu'un déterminant n'est jamais précédé d'un mot modifiant le nom tête du groupe nominal, voir figure 1.10. Ceci permet de bloquer l'analyse d'un groupe nominal comme (35).

(35) *petit le chat*

1.1.5 Propriétés formelles

L'opération d'adjonction rend les grammaires LTAG légérement dépendantes du contexte (*mildly context-sensitive*) c'est-à-dire plus puissantes que les grammaires hors contexte tout en restant strictement inclues dans la classe des grammaires contextuelles.

Le tableau 1.1 résume les propriétés formelles des grammaires légérement dépendantes du contexte, dont font partie les TAG par rapport aux grammaires régulières, hors contexte (GPSG est équivalent à une telle grammaire), et aux grammaires contextuelles (LFG est

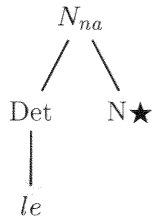


FIG. 1.10 – *Arbre élémentaire auxiliaires associés aux déterminants.*

au pire des cas équivalent à une grammaire contextuelle et HPSG à une machine de Turing, donc un langage non contraint). Notons que les langages réguliers permettent de reconnaître les énoncés non ambigus (plats), les grammaires hors contexte permettent la reconnaissance des énoncés ambigus dont la structure arborescente des dérivations présente des enchassements, les grammaires contextuelles peuvent couvrir tout type de croisement (comme les dépendances croisées en hollandais).

classe de langage	reconnaisseur	complexité
régulier	FSA	$\mathcal{O}(n)$
non contextuel	PDA	$\mathcal{O}(n^3)$
légerement contextuel	2SA	$\mathcal{O}(n^6)$
contextuel	machine de Turing	exponentiel

TAB. 1.1 – *Tableaux des complexités selon la classe de langage.*

Étant donné pour les systèmes de dialogue un objectif d'application automatique proche du temps réél, les applications existantes se restreignent à des algorithmes de traitement polynômiaux, en général des grammaires contextuelles. L'application de grammaires légérement dépendantes du contexte pour ce style d'applications est encore un domaine de recherche.

Les algorithmes d'analyse syntaxique existant pour les grammaires hors contexte sont transposables aux grammaires légèrement dépendantes au contexte, donc aux TAG. Les algorithmes de type CKY (Cocke-Kasami-Younger), LR(0) et Earley ont été réinterprétés pour les grammaires TAG. Le tableau 1.2, repris de [Abeille91b], résume d'une manière plus précise la complexité pour l'analyse d'une grammaire hors contexte, d'une grammaire TAG et d'une grammaire contextuelle.

	grammaires hors contextes	grammaires TAG	grammaires contextuelles
temps	G^2n^3	G^2n^6	exponentielle
espace	Gn^2	Gn^4	exponentielle

TAB. 1.2 – *Tableaux des complexités d'analyse au pire des cas spatiales et temporelles selon le type de grammaire.*

L'analyse syntaxique basée sur des grammaires TAG est donc plus coûteuse que pour les grammaires hors contexte, le temps d'analyse est au pire des cas proportionnel à la puissance 6 de la longueur n de la phrase, et au carré de la taille G de la grammaire. En pratique, le problème essentiel de la complexité des LTAG n'est pas uniquement la complexité en n, l'opération amenant une telle complexité au pire des cas en n étant l'adjonction englobante est en pratique très rare. Il faut tenir compte du fait que le traitement automatique des langues naturelles s'applique en pratique à des phrases assez courtes (surtout par rapport aux langages formels de programmation), et utilise des grammaires de grande taille. Ce point est d'autant plus sensible lorsque la grammaire est lexicalisée, puisqu'elle présente alors une forte redondance de sous-structures, ou de règles, projetées par les *items* lexicaux.

D'autre part, les grammaires TAG associées à des structures de traits (*Feature-Based Tree Adjoining Grammars*) présentent cette fois une complexité spatiale et temporelle exponentielle, comme expliqué précédemment (traits atomiques *ré-entrants*). Ceci est à relativiser, car les TAG ont un emploi plus simple du mécanisme d'unification par rapport aux grammaires basées sur des règles de réécritures, grâce à la localité étendue. Le point de vue est ici différent : les formalismes comme LFG ou HPSG où le squelette hors contexte est là pour filtrer les structures clairement impossibles, les unifications étant là pour réellement contrôler les constructions. À l'inverse pour le formalisme LTAG, le squelette structurel arborescent assume l'essentiel des contrôles sur les constructions licites ou non, tandis que les traits ont plutôt un rôle de filtrage. Il est important également de préciser que l'analyse structurelle reste plus efficace, informatiquement parlant, que les mécanismes d'unification qui imposent un grand nombre de recopies des structures de traits. Même avec de bonnes implantations d'algorithmes optimisés cherchant à minimiser les copies comme [Pereira85], les analyseurs conçus pour des grammaires syntagmatiques à large couverture et basées sur l'unification, comme HPSG, passent autour de 85 à 90% de leur temps à unifier et à copier des structures de traits [Tomabechi91], et peuvent allouer de un à deux MegaOctets de zone mémoire pour l'analyse de simplement huit mots (Flickinger). L'intérêt d'une limitation à des traits à valeurs atomiques et de se reposer au maximum sur la composante structurelle semble donc un bénéfice important du formalisme LTAG.

Certains travaux ont tenté de voir le formalisme TAG et les grammaires légèrement dépendantes du contexte [Weir88] comme une généralisation des grammaires hors contextes. Dans [Boullier98], les TAG sont vues comme des grammaires RCG (Range Concatenation Grammars) particulières, le formalisme des RCG offrant un continuum dans lequel s'inscrivent incrémentalement la plupart des formalismes existants et allant au-delà des langages légèrement dépendants du contexte.

Avec la possibilité de mener des analyses de complexité polynomiale avec ces différentes propriétés algorithmiques, nous avons néanmoins pu mettre en avant certains des aspects de ce formalisme particulièrement intéressants pour les applications informatiques. Une analyse exponentielle amène à des espaces de recherche incompatibles avec des analyses en temps réel si on souhaite pouvoir couvrir toutes les ambiguïtés. Nous allons maintenant nous intéresser à l'adéquation du formalisme pour la modélisation de la syntaxe du langage naturel.

1.2 LTAG : description du modèle de la langue

L'intérêt linguistique spécifiques des TAG repose essentiellement sur les deux points suivants :

- formalisme mathématique contraint : le pouvoir génératif du modèle va au-delà de celui d'une grammaire hors contexte (reconnue comme trop limité [Shieber85]) mais est plus contraint que les extentions de CFG reposant sur des structures de traits (HPSG, LFG, etc.) ;
- principe de localité étendu : il permet d'exprimer aisément des contraintes sur les compléments, les sous-catégorisations et les expressions idiomatiques.

Le dernier point est spécifique aux formalismes fondés sur des réécritures d'arbres plutôt que sur des réécritures de chaînes. Cette propriété permet de représenter de manière simple les problèmes de sous-catégorisation, les problèmes de contraintes sur différentes parties d'un arbre (comme les accords sujet-verbe) et les expressions idiomatiques.

Trois principes linguistiques de bonne formation des arbres élémentaires ont été définis par [Kroch et al.85] et [Abeille91a]. Nous allons les présenter et tenter de les justifier dans les sections suivantes.

1.2.1 Lexicalisation

La lexicalisation repose sur le premier principe suivant :

> **Principe 1 : lexicalisation intégrale de la grammaire**
> Tout arbre élémentaire comporte au moins un nœud feuille lexical non vide [Schabes et al.90a][Abeille91b].

Dans [Schabes et al.90a], les auteurs présentent le principe de lexicalisation d'une grammaire TAG et les avantages linguistiques et informatiques.

Sur le plan linguistique, la lexicalisation d'un formalisme rend compte que le comportement syntaxique dépend directement du mot employé et ne peut se généraliser par l'emploi d'un jeu restreint de catégorie. La lexicalisation est une propriété moins spécifique au modèle, puisque LFG et HPSG tentent d'en exploiter aussi les avantages, mais fondamentale pour l'expressivité d'un formalisme.

Pour les applications informatiques, l'analyse d'une phrase ne se réalisera plus en considérant l'intégralité de la grammaire, mais simplement le sous-ensemble des arbres élémentaires ancrant un ou plusieurs mots de la phrase.

1.2.2 Localisation et caractérisation des dépendances

La localisation et la caractérisation des dépendances sont établies pour les LTAG essentiellement par le principe suivant :

> **Principe 2 : cooccurrence prédicat-argument**
> Tout arbre élémentaire ancré par un prédicat comporte au moins un nœud pour chacun des arguments que le prédicat sous-catégorise [Abeille91b].

Ce principe est rendu possible par la propriété de domaine de localité étendu. L'intérêt fondamental pour l'analyse est que toutes les dépendances peuvent être indiquées directe-

ment dans la structure des arbres élémentaires et qu'elles n'ont pas besoin d'être calculées dynamiquement durant l'analyse par le biais d'unification.

Le principe de localité entendue combiné à la lexicalisation de la grammaire induit cependant un problème important, mais souvent ignoré, pour une mise en œuvre informatique, celui de la redondance des sous-structures communes. Nous verrons cependant que certaines techniques comme la précompilation des actions de l'analyseur sous forme de table d'analyse ou la précompilation de la grammaire sous forme d'un automate apportent une réponse tout à fait satisfaisante à ce problème.

Les choix linguistiques définis dans [Abeille91b] imposent un certain nombre de critères pour déterminer quel type d'arbre associer à la représentation d'un phénomène particulier. Les arbres élémentaires peuvent être de deux types :

- les arbres *initiaux* sont utilisés pour la représentation des noms communs ou propres, des verbes intransitifs ou transitifs à complément nominal ou prépositionnel ;

- les arbres *auxiliaires* servent à représenter des modifieurs (adjectifs, adverbes, relatives), des verbes à complétives, des verbes modaux et des auxiliaires.

Afin de préciser quel type d'arbre doit être utilisé pour quelle catégorie, [Candito99] apporte des principes complémentaires. Ces choix permettent d'identifier le type de dépendances mis en jeu lors d'une dérivation.

1.2.3 Construction des arbres élémentaires

Le principe suivant permet de définir les contraintes de formation des arbres élémentaires relativement à un choix linguistique fort qui est celui d'une correspondance directe entre l'ensemble des ancres d'un même arbre élémentaire et un lexème, c'est-à-dire une unité sémantique :

> **Principe 3 : minimalité sémantique**
> L'ensemble des ancres de tout arbre élémentaire correspond à exactement une unité sémantique non vide [Abeille91b][Candito99].

Ce principe implique qu'un arbre élémentaire ne peut pas être ancré seulement par un mot fonctionnel, c'est-à-dire sémantiquement non autonome [Abeille91b]. Le ou les ancres d'un même arbre élémentaire doivent correspondre à une flexion d'un lexème [Candito99]. Ainsi combiné aux deux principes précédents, les verbes prédicatifs et les prépositions régies par le verbe apparaissent dans un même arbre élémentaire, comme illustré sur la figure 1.11.

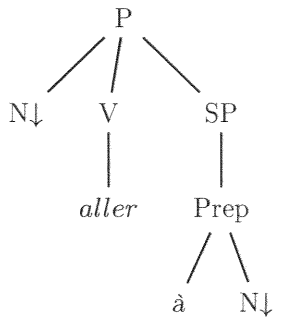


FIG. 1.11 – *Arbre élémentaire associé au prédicat aller avec arguments nominaux sujet et objet, ce dernier étant introduit par la préposition à.*

Les expressions figées (c'est-à-dire les expressions dont le sens est non-compositionnel) sont de même représentées par des arbres élémentaires contenant autant d'ancres que de mots constituant l'expression. C'est aussi le cas des verbres supports qui doivent apparaître avec le nom prédicatif comme pour l'exemple 1.12.

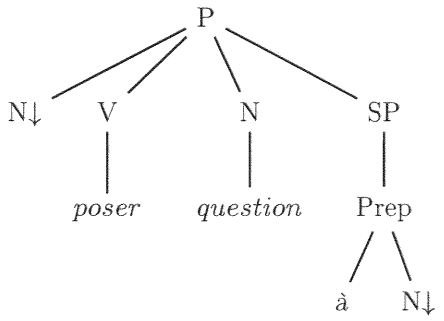


FIG. 1.12 – *Arbre élémentaire associé au prédicat figé questionner construit avec verbe support.*

Dans le cas où un arbre élémentaire présente plusieurs ancres, on en distingue une, appellé *ancre principale*, qui correspond à l'ancre dominante syntaxiquement (le verbe par exemple dans les constructions verbales). Les autres ancres sont appellées *co-ancres* ou *co-têtes*.

1.2.4 Interprétation linguistique des résultats d'analyse

L'arbre dérivé, comme nous l'avons indiqué précédemment ne permet pas de déduire de manière univoque le processus de composition syntaxique d'un énoncé. Il ne permet donc pas de représenter les ambiguïtés sémantiques, ce qui est le rôle de l'arbre de dérivation.

On obtient en effet :

- pour une phrase ambiguë syntaxiquement (ou structurellement), autant d'arbres dérivés et de dérivation que d'analyses possibles ;

- pour une phrase ambiguë sémantiquement, un arbre dérivé et autant d'arbres de dérivation que d'interprétations sémantiques possibles.

Nous pouvons illustrer le second cas de figure en prenant l'exemple de l'analyse de la phrase *Jean prend une veste*, figure 1.13 avec l'ambiguïté de l'interprétation compositionnelle (prendre un vêtement) et de l'interprétation idiomatique semi-figée (se faire ridiculiser).

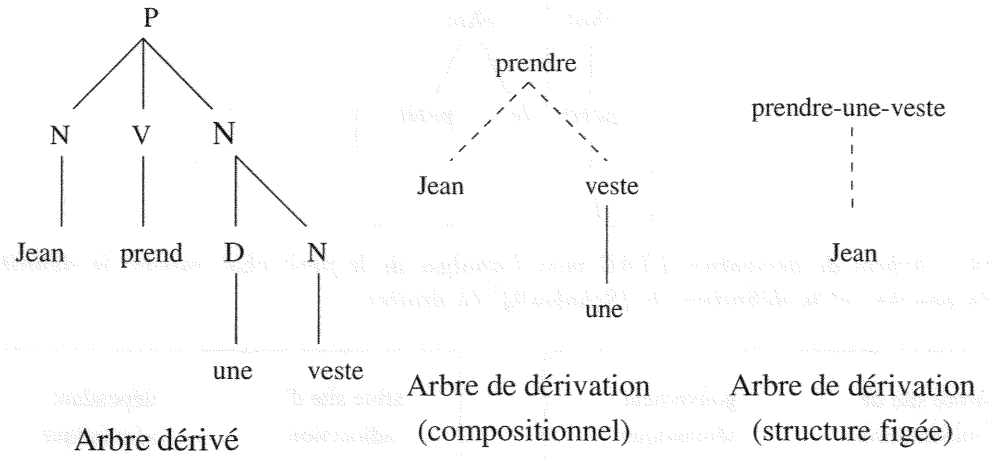


FIG. 1.13 – *Analyse de la phrase Jean prend une veste*

Grâce au principe 3 de minimalité sémantique, les arbres élémentaires correspondent à des unités sémantiques, de plus, selon le principe 2 de coocurrence prédicat-argument, chaque arc de l'arbre de dérivation correspond à une dépendance prédicative. L'arbre de dérivation correspond alors une description proche d'un arbre de dépendance, et constitue ainsi une bonne base pour l'analyse sémantique. La définition des dérivations de [Schabes94] autorisant les adjonctions multiples sur les nœuds et distinguant adjonctions prédicatives et modifieurs permet d'obtenir un arbre de dérivation plus proche des dépendances sémantiques comme le montre l'exemple de la figure 1.14 pour l'analyse de *le petit chat*.

En se basant sur la notion de dépendance syntaxique et sémantique de la Théorie Sens-Texte [Melcuk88] et sur la définition des dérivations de [Schabes94], [Candito et al.98] propose d'interpréter l'arbre de dérivation en tant que graphe des dépendances sémantiques (les coréférences ne sont pas prises en compte). Chaque nœud de l'arbre de dérivation est traduit par le sémantème signifié par le lexème qui ancre l'arbre élémentaire[34]. L'interprétation des arcs de l'arbre de dérivation comme dépendances sémantiques est alors donnée par la figure 1.15.

Si nous prenons l'exemple de la phrase (36) tirée de [Candito99], en considérant une grammaire LTAG construite selon les principes présentés et la définition de [Schabes94], nous obtenons l'arbre de dérivation et le graphe des dépendances sémantiques de la figure 1.16.

34. Certains nœuds correspondants à des flexions (comme les auxiliaires de temps) peuvent être incorporés comme traits [Candito99]. D'autre part les déterminants ne sont pas considérés comme réalisation de dépendances sémantiques et sont supposés être pris en compte par un mécanisme supplémentaire pour le traitement de la quantification.

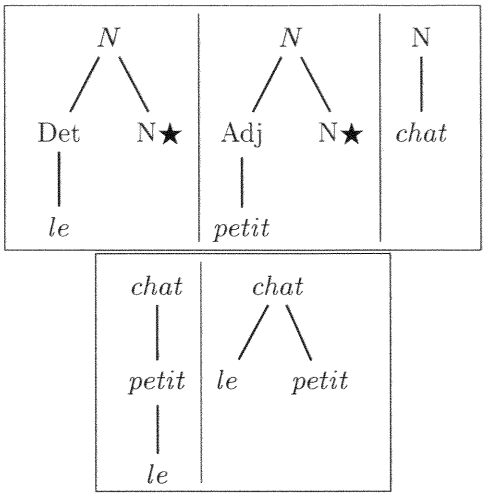


FIG. 1.14 – *Arbres de dérivation LTAG pour l'analyse de le petit chat suivant la définition de [VS87] (à gauche) et la définition de [Schabes94] (à droite).*

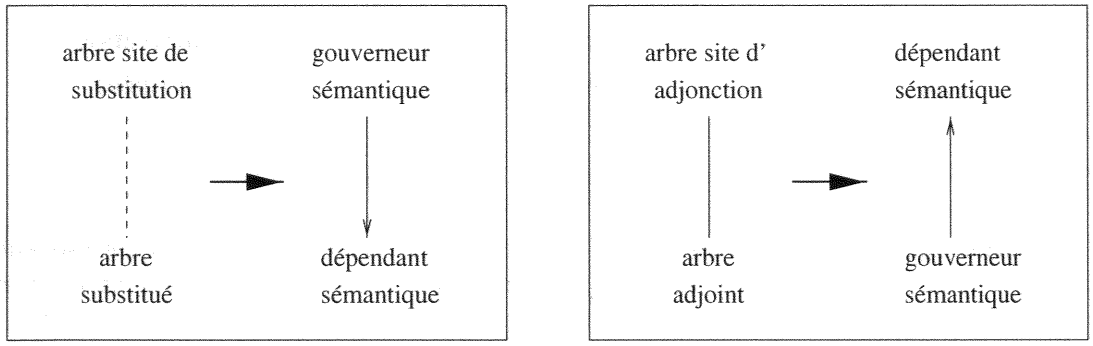


FIG. 1.15 – *Interprétation des arcs de l'arbre de dérivation comme des dépendances sémantiques selon [Candito99].*

(36) *La nouvelle bibliothèque possède le livre que Marie pense que Jean cherche.*

1.2.5 Un niveau supplémentaire de description : la métagrammaire

Nous avons présenté les principes linguistiques classiquement associés au formalisme des LTAG. Ces principes permettent d'écrire des arbres élémentaires et d'interpréter les résultats fournis par l'analyse syntaxique. Cependant, le formalisme ne fournit pas de mécanisme permettant d'exprimer des généralisations syntaxiques partagées entre plusieurs arbres élémentaires, comme par exemple l'accord sujet-verbe. D'une façon générale, bien que fortement lexicalisé, le formalisme des LTAG ne s'accompagne pas d'une organisation générale des informations linguistiques comparables, par exemple, aux principes de typage et d'héritage des grammaires HPSG. Une grammaire LTAG à largre couverture d'une langue peut compter plusieurs milliers d'arbres élémentaires. L'écriture d'une telle grammaire amène vite des problèmes de cohérence que ce soit au niveau de l'utilisation des traits, des catégories ou du croisement de phénomènes linguistiques.

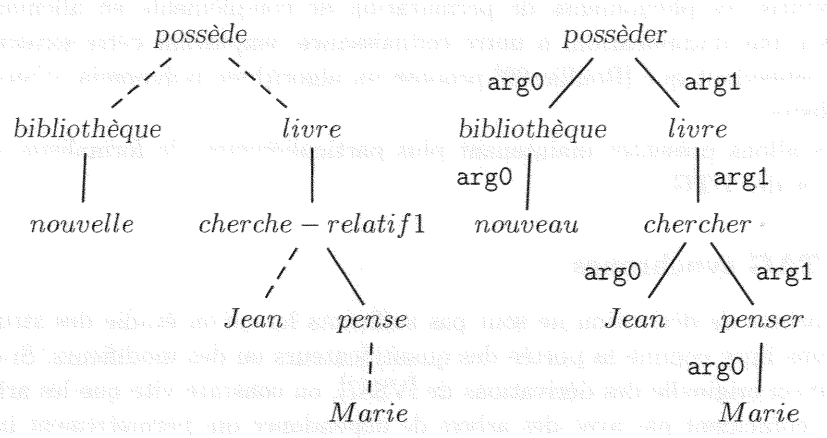


FIG. 1.16 – *Arbre de dérivation LTAG et graphe des dpendances sémantiques pour l'analyse de (36) selon [Candito99]*

Différentes organisations des grammaires LTAG ont tenté de répondre à ces problèmes. La difficulté est de parvenir à une représentation compacte d'une grammaire LTAG offrant à la fois une vision globale de la grammaire et un niveau d'expressivité satisfaisant, linguistiquement motivé, pour l'écriture de cette grammaire. [VS et al.92] ont proposé tout d'abord une organisation fondée sur un graphe d'héritage monotone. [Becker93] a proposé un système à base de métarègles pour l'allemand. Enfin [Evans et al.95] ont développé une compilation de grammaire LTAG à l'aide du formalisme DATR sous la forme d'un graphe d'héritage non monotone incluant des règles lexicales. Les propositions de [VS et al.92] ont été redéveloppés par [Candito99] qui propose un cadre de description linguistique plus abstrait et plus modulaire, la métagrammaire.

La métagrammaire consiste en trois hiérarchies qui correspondent respectivement à la sous-catégorisation initiale, aux redistributions et aux réalisations des fonctions grammaticales. Un mécanisme d'héritage monotone permet de générer l'ensemble des arbres élémentaires non instanciés d'une grammaire LTAG en assurant la bonne formation des structures lexicalisées. Ce niveau supplémentaire de description permet d'exprimer de manière souple les généralisations syntaxiques, facilitant ainsi la maintenance d'une LTAG, tout en assurant une méthode effective de compilation de la grammaire.

1.3 Extensions du formalisme

A partir des limites linguistiques de représentation constatées, des extensions du formalisme ont tenté d'en augmenter sa puissance et son expressivité. Citons tout particulièrement :

- les TAG ensemblistes (*multi-component TAG*) [Weir88] ;
- les TAG synchrones [Shieber et al.90] ;
- le formalisme des Grammaires de Descriptions d'Arbres (DTG) [Rambow94].

Dans l'extension des TAG ensemblistes, les structures élémentaires sont des ensembles d'arbres qui doivent être utilisés simultanément lors d'une combinaison avec une autre structure. Si on autorise les arbres élémentaires d'un même ensemble à se combiner avec des arbres différents, on obtient un modèle plus puissant que les TAG ordinaires aptes

à représenter les phénomènes de permutation de compléments en allemand [Becker91]. Il n'existe pas d'application, à notre connaissance, employant cette extention des TAG. Notons cependant que [Boullier99] propose un algorithme polynomial d'analyse des TAG ensemblistes.

Nous allons présenter maintenant plus particulièrement le formalisme des TAG synchrones et des DTG.

1.3.1 Les TAG synchrones

Les arbres de dérivation ne sont pas suffisants lorsqu'on étudie des structures sémantiques plus fines comme la portée des quantificateurs ou des modifieurs. Si on s'en tient à la définition originelle des dérivations de [VS87], on constate vite que les arbres de dérivations ne coïncident pas avec des arbres de dépendance qui permettraient immédiatement une analyse sémantique.

Afin de combler les limites présentées par l'arbre de dérivation, Stuart Shieber et Yves Schabes proposèrent dans [Shieber et al.90] le mécanisme des TAG synchrones, qui permet d'obtenir une ou plusieurs représentations sémantiques autonomes basées également sur les TAG, c'est-à-dire sous la forme d'une grammaire arborescente. Outre l'interface entre syntaxe et sémantique, les TAG synchrones ont également été utilisées pour la traduction automatique (la synchronisation portant sur deux TAG syntaxiques décrivant deux langues différentes), pour la génération [Shieber et al.92] et pour l'organisation de grammaire LTAG [Abeille91b]. Les TAG synchrones permettent en particulier de récupérer les dépendances sémantiques correctes tout en fonctionnant selon la définition de [VS87].

La synchronisation de deux grammaires TAG consiste à coupler les arbres de la première grammaire avec ceux de la seconde en spécifiant des correspondances entre les structures élémentaires et les processus de construction des arbres. La définition originale présentée dans [Shieber et al.90] est la suivante : une grammaire TAG synchrone est donnée par un ensemble de triplets (L_i,R_i,s_i), où les ensembles d'arbres élémentaires $\{L_i\}$ et $\{R_i\}$ forment respectivement deux grammaires TAG standards et où s_i est une fonction établissant des liens entre des adresses de nœuds de L_i et de R_i. On associe un processus de réécriture durant l'analyse qui consiste à répéter les trois points suivants jusqu'à dérivation complète d'un énoncé considéré [Abeille93] :

- choisir une première paire d'arbres élémentaires (E_1,E_2), un nœud A de E_1 et le nœud B de l'arbre E_2 lié à A par la synchronisation ;
- choisir une autre paire d'arbres (E_3,E_4), E_3 pouvant s'adjoindre ou se substituer en A de E_1, E_4 pouvant s'adjoindre ou se substituer en B de E_2 ;
- combiner E_3 en A et E_4 en B, ce qui donne une nouvelle paire d'arbres dérivées, et supprimer le lien entre A et B.

Cette définition permet en particulier la synchronisation de deux analyses correspondant à des dérivations non-isomorphes courantes pour le couplage entre syntaxe et sémantique. Cependant cette définition rend également possible la génération de langages qui ne sont plus des langages d'arbres adjoints. L'augmentation de la puissance générative conséquente à cette définition rend impossible l'implantation d'un analyseur de TAG synchrone efficace. Afin de rester dans la classe des langages d'arbres adjoints, [Shieber94] propose de contraindre la synchronisation par un isomorphisme des arbres de dérivations correspondant aux deux TAG. Le formalisme devient alors informatiquement utilisable, mais ne couvre plus certains phénomènes de traduction qui imposent un non isomorphisme des dé-

rivations. [Rambow et al.96] propose une troisième définition qui constitue un compromis plus contraint que la solution initiale, mais dont la puissance formelle va toutefois au-delà de la classe des langages d'arbres adjoints.

1.3.2 Les DTG

Le formalisme des DTG (D-Tree Grammars), Grammaires de Description d'Arbres, reprend les avantages des TAG tout en allant au-delà de certaines de ses limites d'ordre linguistique ; il est présenté dans [Rambow et al.95a] et constitue le résultat des recherches formelles de trois anciens élèves de A. Joshi. Il est issu tout d'abord des travaux de Vijay-Shanker sur les descriptions d'arbres (*quasi-arbres*) [VS93], qui ont pour origine l'application de la *D-theory* de Marcus pour les TAG. L'objectif de ce travail était de redéfinir une grammaire TAG comme un ensemble de structures arborescentes où les relations de dominance sont partiellement spécifiées. Les nœuds internes et racines sont ainsi considérés comme doubles, leurs deux composantes étant liées par un chemin de longueur sous spécifiée pouvant être nulle (dominance alors immédiate). L'intérêt est alors que l'opération d'adjonction dans ce nouveau cadre est monotone. Les DTG sont aussi inspirées des travaux d'Owen Rambow qui a défini une extension du formalisme nommé V-TAG pour la prise en compte des langues à ordre de mots libres, d'origines aussi diverses que l'allemand, le coréen ou encore le kashmir [Rambow94]. La troisième source d'inspiration est le travail de David Weir sur les formalismes légèrement dépendant du contexte, sur la caractérisation des descriptions structurelles des formalismes existants et sur la modélisation des phénomènes qui échappent à des grammaires LTAG classiques [Weir88].

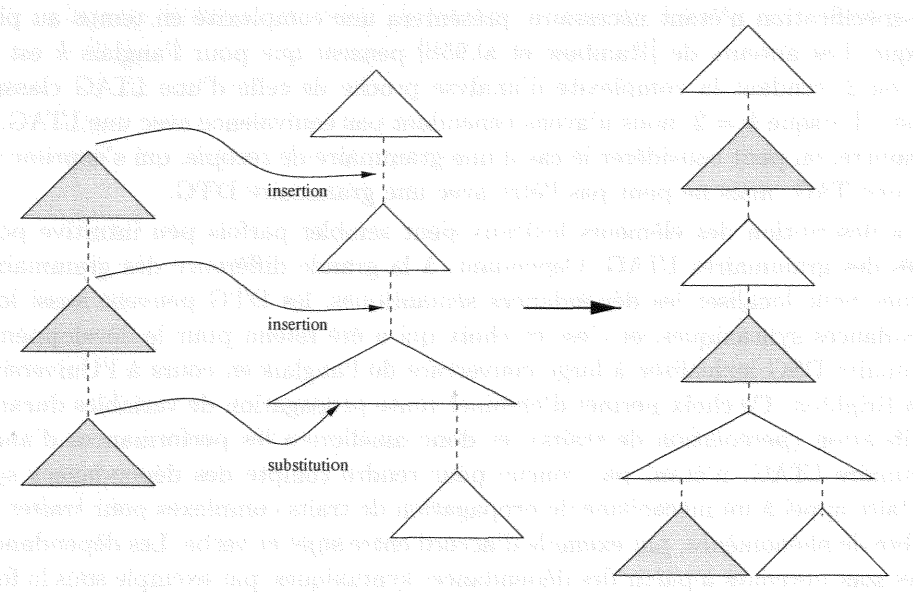


FIG. 1.17 – *Un exemple d'opération de subsertion (ou substitution généralisée) pour les DTG (maintenant DSG).*

Deux opérations permettent de combiner des unités lexicales qui sont des arbres élémentaires correspondant à des quasi-arbres. La première de ces opérations est la *subsertion* qui généralise la substitution (voir figure 1.17), rend inutile l'adjonction prédicative et correspond toujours au rattachement prédicat-argument. L'idée de cette opération est

d'autoriser certaines parties de l'arbre substitué à s'insérer dans l'arbre site de la substitution. La seconde est l'*adjonction sœur*, sorte d'adjonction plate rappellant l'opération de furcation [Smedt et al.90], qui correspond toujours au rattachement d'un modifieur. Notons que dans les dernières présentations du formalisme cette dernière opération disparait, que le formalisme se trouve renommé en DSG (D-Tree Substitution Grammars) et la *subsertion* en substitution. En effet, l'adjonction plate utilisée n'était pas sans poser problèmes pour des raisons présentées dans [Candito99], et il est alors possible de se ramener à une unique opération de substitution généralisée. Le principal intérêt du formalisme tient en ce que la structure prédicat-argument correcte est toujours déduite directement de l'arbre de dérivation, qui constitue alors une base immédiate pour une analyse sémantique. Ceci a par exemple été expérimenté à l'aide de λ-terme comme présenté par [Hepple99] pour le traitement de quantifieur. De plus le formalisme permet d'obtenir des analyses uniformes pour les dépendances à longue distance, les *wh-movement*[35] en anglais et les rattachements corrects pour les langues à ordre libre de mots.

D'un point de vue formel, les DTG peuvent être analysées polynomialement, leur complexité varie en fonction du nombre de sous-spécifications associées aux arbres élémentaires : chaque sous-spécification dans un arbre élémentaire correspond, lors d'une subsertion, à une possibilité supplémentaire de combinaison de sous-chaînes non adjacentes, donc à une complexité supplémentaire du processus de reconnaissance sur la chaîne d'entrée. La complexité d'analyse en temps au pire des cas d'une DTG est en $\mathcal{O}(n^{4k+3})$ pour l'algorithme de type Earley présenté dans [Rambow et al.95a], où n est la longueur de la chaîne à analyser et k le nombre maximal de sous-spécifications possibles dans un arbre élémentaire. En particulier, donc, une grammaire DTG lexicalisant une grammaire hors contexte, aucune sous-spécification n'étant nécessaire, présentera une complexité en temps au pire des cas cubique. Les auteurs de [Rambow et al.95b] pensent que pour l'anglais k est de l'ordre de 1 ou 2, rendant la complexité d'analyse proche de celle d'une LTAG classique, donc réaliste. Lorsque $k = 2$, nous n'avons cependant pas équivalence avec une LTAG. Pour s'en convaincre, on peut considérer le cas d'une grammaire *de recopie*, qui s'exprime facilement avec une TAG, mais ne peut pas l'être avec une grammaire DTG.

La description des éléments lexicaux peut sembler parfois peu intuitive pour les habitués des grammaires LTAG. Cependant, à la grande différence des grammaires LTAG, conçues pour localiser les dépendances sémantiques, les DTG peuvent aussi localiser les dépendances syntaxiques, et c'est ce choix qui a été retenu pour le développement d'une grammaire DSG lexicalisée à large couverture de l'anglais en cours à l'Université du Sussex à Brighton. Ce choix permet d'éliminer toute propagation de variables durant la phase d'unification (*percolation* de traits), et donc améliorera les performances d'analyse. Une grammaire LTAG, n'étant pas conçue pour rendre compte des dépendances syntaxiques doit faire appel à un mécanisme de propagation de traits complexes pour traiter un certain nombre de phénomènes, par exemple d'accord entre sujet et verbe. Les dépendances sémantiques sont obtenues à partir des dépendances syntaxiques, par exemple sous la forme d'une forêt de dérivations, en faisant appel à des mécanismes de mouvements des constituants.

Il faut noter que le coût d'analyse structurelle d'une DTG peut se révéler potentiellement fort dû aux sous-spécifications et à l'importance de l'espace de recherche que celà peut amener[36]. Sur ce dernier point, l'application de méthodes plus systématiques pour la

35. C'est-à-dire déterminer les dépendances correctes dans le cas de pronoms relatifs et interrogatifs.

36. L'augmentation du nombre d'arbres élémentaire DTG observé par [Smets98] par rapport au LTAG semble plus tenir des choix linguistiques employés dans la grammaire en question que sur une propriété particulière du

construction d'analyseur, sur le principe de celle proposée par [Billot et al.89] et [Lang91] qui passe par la construction d'un PDA[37] ou d'une LIG équivalent pour représenter les étapes de reconnaissance, semble particulièrement bien adaptée. Cette approche est en cours de développement par les concepteurs du formalisme afin d'obtenir des analyseurs de type Earley et CKY pour les DTG. De manière générale, ce formalisme est une amélioration notable du formalisme LTAG, rendant l'arbre de dérivation très proche d'un arbre de dépendance syntaxique ou sémantique, voir équivalent avec l'extension proposée dans [Candito et al.98] (grammaires GAG) et en considérant la définition du graphe des dépendances sémantique de la Théorie Sens-Texte, et apte à représenter certains phénomènes (*scrambling* en allemand et coréen par exemple) qui constituaient des limites aux TAG.

1.3.3 TAG et linguistique : conclusion

Par sa capacité générative faible et forte, par les travaux en linguistique réalisés, le formalisme des LTAG constitue l'un des plus intéressants pour des applications informatiques. Ceci s'exprime tout d'abord en tant que formalisme descriptif : la conception d'une grammaire repose sur des principes précis et les unités élémentaires restent très proches des arbres syntagmatiques traditionnels en linguistique. Le domaine de localité étendu permet d'exprimer simplement et directement dans la partie structurelle des contraintes qui auraient nécessité des équations de traits réentrants complexes et sans doute moins lisibles avec des formalismes comme LFG et HPSG. Ceci s'exprime aussi en tant que formalisme d'analyse, puisque des analyses polynomiales restent possibles. Ce n'est pas un hasard si, sur ce dernier point, certains ont vu dans les TAG un formalisme intéressant pour compiler efficacement des grammaires HPSG [Kasper et al.95] ou GB.

Nous allons maintenant nous intéresser aux applications pratiques qui ont été menés ces dernières années sur la base de ce formalisme.

formalisme.

37. Push Down Automaton.

Chapitre 2

Applications informatiques des LTAG

Les différents intérêts du formalisme LTAG ont conduit à de nombreux développements informatiques. Encore maintenant il s'agit sous doute avec HPSG du formalisme grammatical donnant lieu au plus grand nombre de recherches en traitement automatique[38]. Cette étude des applications informatiques repose sur trois principaux points : tout d'abord une présentation des principales implantations existantes du formalisme et d'une de ses variantes (XTAG, FTAG et le projet LexSys). D'autre part, le développement de grammaires à large couverture de la langue s'est accompagné d'études de modèles probabilistes et enfin de la mise en œuvre de versions *simplifiées* du formalisme LTAG.

2.1 Réalisations informatiques

2.1.1 XTAG

Entrepris dès 1987, le système XTAG [rg98] est l'implantation la plus ambitieuse du formalisme LTAG[39]. Le système se compose d'un analyseur morphologique reposant sur une base de données morphologique de l'ordre de 317000 formes fléchies (90000 lemmes). Un étiqueteur automatique à base de trigrammes et entrainé sur un Corpus du Wall Street Journal (1 million de mots) constitue une première étape de désambiguïsation et de réduction de l'espace de recherche futur. Un second étiqueteur, de structures syntaxiques cette fois (*supertaggueur*, technique que nous présenterons plus en détail dans la section suivante), offre une seconde passe de désambiguïsation avant analyse. Un lexique syntaxique rassemble 105000 entrées apportant pour un lemme donné la liste des familles d'arbres valides et des contraintes lexicales sous forme de traits. Une banque de 886 arbres élémentaires non instanciés est répartie en 48 familles d'arbres. XTAG inclut enfin une interface graphique permettant la création, la maintenance et des tests d'analyse.

L'ampleur de cette réalisation constitue une référence difficilement contournable. Il faut cependant préciser les limites présentées par le système XTAG et qui compromettent son utilisation pour nos objectifs :

- conditions d'installation défiant les meilleurs volontés ;
- difficulté d'extension du système ;

38. On pourra se référer par exemple aux sessions spéciales TAG de la conférence COLING de 1998.

39. Il s'agit également d'une des implantations dans le domaine de l'analyse syntaxique ayant impliqué le plus de participants (de l'ordre de 35 personnes) ayant travaillé directement à l'élaboration de ce système, et au moins autant de façon plus périphérique [Doran et al.94].

– limites en matière de portabilité (format propriétaire).

2.1.2 FTAG

En développant une grammaire du français pour le système XTAG, Anne Abeillé a réalisé le premier système d'analyse du français reposant sur une grammaire d'arbres adjoints [Abeille91b], [Abeille et al.94], complété par les travaux de Maris-Hélène Candito [Candito99]. D'importants efforts ont été mis en œuvre pour réaliser une grammaire à large couverture, si bien qu'il s'agit, avec par exemple [Wehrli97], de l'un des systèmes existants d'analyse du français le plus important à l'heure actuelle. La grammaire actuelle contient de l'ordre de 5000 schèmes grâce à l'enrichissement permis par le système de [Candito99]. Notons cependant que certains de ces schèmes ne se distinguent que par les traits.

Là encore, les mécanismes d'unification étant locaux, le système de traits utilisé dans la grammaire du français est plus simple que ceux utilisés dans les autres formalismes d'unification. D'autre part, pour améliorer l'organisation de la grammaire, la notion de familles d'arbres élémentaires a été développée : l'ensemble des arbres élémentaires qui correspondent aux différentes structures syntaxiques de surface associées à un même prédicat donné constitue une famille d'arbres. Ainsi, un arbre appartenant à la famille de « N0 aimer N1 » est par exemple celui correspondant au passif (« N1 être aimé par N0 »). Une autre particularité de ce système est l'extension de la couverture de la grammaire à certaines formes idiomatiques.

2.1.3 Le projet LexSys

Le projet en cours LexSys est mené par David Weir à l'Université du Sussex, Royaume Uni. L'objectif de ce projet est de développer une grammaire à large courverture fondée sur le formalisme DTG lexicalisé, variante des LTAG présentée dans le chapitre précédent (maintenant DST pour *D-Tree Substitution Grammar*). Ce vaste et ambitieux projet est aussi l'occasion de tester et d'approfondir quelques unes des techniques les plus récentes du traitement automatique des langues sur ce style de grammaire :

– une organisation de la grammaire fondée sur une hiérarchie et un héritage non-monotone ;
– l'exploitation de techniques de pré-compilation pour améliorer l'analyse syntaxique ;
– l'utilisation d'analyses statistiques pour desambiguïser les résultats d'analyse.

Ces techniques s'appuient sur des outils et des ressources existantes comme le lexique développé dans le cadre du projet *Alvey*[40], des informations sur les fréquences lexicales du projet SPARKLE[41] ou le formalisme de représentation des connaissances lexicales DATR. Il s'agit en soi d'un excellent exemple de réinvestissement de ressources existantes et qui montre tout l'intérêt de liens forts entre une communauté scientifique nationale.

Ce projet permet de valider d'une façon pratique l'adéquation du formalisme DTG pour la description de l'anglais et les avantages par rapport à son frère ainé TAG. Il confronte aussi le formalisme aux questions de mise en œuvre automatique. [Smets98] note

40. Le projet *Alvey Natural Language Tools* [Briscoe et al.87] avait également pour objectif le développement d'une grammaire à large couverture pour l'angais fondé sur GPSG et de divers outils pour l'analyse et la gestion des ressources associées.

41. SPARKLE : Shallow PARsing and Knowledge extraction for Language Engineering, voir `http://www.ilc.pi.cnr.it/sparkle.html`

que le nombre d'arbres élémentaires pour cette implantation est supérieur à celui observé dans XTAG, bien que les phénomènes couverts soient du même ordre. Cette observation semble liée aux choix linguistiques suivis, mais également à des efforts pour améliorer les performances d'analyse pour de tels formalismes fortement lexicalisés, basés sur des précompilations de la grammaire et sur la désambiguïsation probabilistes. De plus, la nature même du formalisme et le choix de la représentation des dépendances syntaxiques plutôt que sémantiques comme pour les grammaires LTAG, simplifie la phase d'unification qui peut être menée efficacement durant l'analyse [Carroll et al.99a].

2.2 Intérêts pour l'analyse robuste

Outre les développements de grammaires à large couverture de la langue et de divers outils associés, le formalisme LTAG a pu bénéficier d'un nombre important de travaux en vue d'améliorer les performances et la robustesse des analyses menées.

Srinivas [Srinivas97a] a souligné certains aspects rendant intéressant l'utilisation de LTAG pour l'analyse de portions de la phrase. Suite à une simple première passe de rattachement des modifieurs nominaux, les résultats obtenus sont proches du premier niveau d'analyse par segments du système CASS de [Abney90].

Compte tenu de l'approche présentée dans la partie précédente, cet aspect nous parait essentiel dans la mesure où il prouve que le formalisme LTAG peut fort bien s'appliquer à des analyses partielles centrées sur les noyaux grammaticaux des énoncés des utilisateurs. D'autre part, la compétence exprimée dans les grammaires LTAG existantes semble tout à fait à même de permettre le déroulement de l'analyse syntaxique selon l'axe syntagmatique défini dans [BB90]. Ceci suppose alors, à notre sens, de définir un algorithme capable d'analyser les segments grammaticaux les plus étendus et d'utiliser des règles de rattrapage supplémentaires dédiées aux variations de l'axe paradigmatique, en vue de rétablir l'axe syntagmatique analysable par la grammaire LTAG. Le cas particulier des ellipses et des effacements, nous l'avons vu, ne pourra être couvert par ces règles paradigmatiques, et devra, à notre sens, être directement modélisé dans la grammaire LTAG.

Nous avons souligné, dans la partie précédente, l'importance de la prise en compte des informations probabilistes dans les modèles d'analyse grammaticale afin de parvenir à des analyseurs efficaces. Nous allons donc maintenant nous intéresser à l'utilisation de probabilité dans le modèle TAG.

2.3 Approches probabilistes

2.3.1 TAG stochastiques

En vue de parvenir à une réduction de l'espace de recherche, des modèles stochastiques associés au TAG ont été développés de façon similaire à ceux des grammaires hors contextes. Deux modèles de TAG stochastiques ont été présentés par [Schabes92] et [Resnik92]. On définit la probabilisation d'une grammaire LTAG de la manière suivante [Rajman95] : si on note I l'ensemble des arbres initiaux, A l'ensemble des arbres auxiliaires et Ω l'ensemble des opérations (substitution S ou adjonction A), une Grammaire d'Arbres Adjoints lexicalisée Probabiliste (SLTAG, Stochastic Lexicalized Tree-Adjoining Grammar) est caractérisée par trois fonctions :

- P_I fonction de I dans [0,1], tel que $\sum_{\alpha \in I} P_I(\alpha) = 1$, P_I représente la probabilité qu'une

dérivation soit une dérivation à partir de l'arbre initial α .

- P_S fonction de Ω dans [0,1], tel que $\forall t \in I \cup A, \forall v \in s(t), \sum_{\alpha \in I} P_S(S(t,\alpha,v)) = 1$, $P_S(t,\alpha,v)$ représente la probabilité de la substitution du nœud v et de l'arbre t par l'arbre initial α.
- P_A fonction de Ω dans [0,1], tel que $\forall t \in I \cup A, \forall v \in a(t), \sum_{\beta \in A} P_A(A(t,\beta,v)) = 1$, $P_A(t,\beta,v)$ représente la probabilité que l'arbre auxiliaire β soit adjoint au nœud v de l'arbre t.

La probabilité d'une dérivation d = (α, $op_1(...)$, ...,$op_n(...)$), où $\forall i, op_i \in \{S,A\}$, s'exprime alors :

$$P(d) = P_I(\alpha) \prod_{i=1}^{n} P_{op_i}(op_i(...))$$

Ceci signifie, comme pour les grammaires stochastiques définies précédemment, que la probabilité d'une dérivation est définie comme le produit des probabilités des règles qui constituent cette dérivation. Les désambiguïsations permises par ce modèle peuvent tout à fait être complémentaires d'un mécanisme particulier d'analyse ascendante par îlots. Un tel modèle permet de définir des principes préferentiels en vue de réduire la combinatoire durant l'analyse. L'analyseur stochastique pourra offrir les dérivations les plus probables, avec un score de confiance associé à chacune de ces hypothèses.

La mise en œuvre de ce modèle reste délicate du fait du grand nombre de paramètres à estimer, demandant des corpus de taille très importante déjà analysés, ressources qui ne sont actuellement pas encore disponibles pour le français.

2.3.2 Modèle de langage et Supertagging

Le point de départ du *supertagging* [Srinivas97a] est de considérer les arbres élémentaires LTAG non instanciés comme des étiquettes employées pour le classique étiquetage morpho-syntaxique. Ces *supertags* constituent des unités lexicales très riches contenant à la fois les catégories syntagmatiques associées aux mots qui les ancrent (*part of speech*), mais également les informations de sous-catégorisation en localisant les dépendances sémantiques. Chaque mot se voit associé avec un ou plusieurs *supertags*, chaque *supertag* représentant un des contextes syntaxiques dans lequel ce mot peut s'employer (ce qui correspond à des informations données dans le lexique syntaxique d'une Grammaire LTAG classique).

Le *supertagging* consiste à sélectionner les *supertags* les plus appropriés pour chaque mot étant donné le contexte de l'énoncé. Le *supertagging* peut être vu comme une étape de désambiguïsation avant analyse, sachant que ce point est essentiel pour une efficacité d'analyse. Pour une grammaire à large couverture, un mot peut se voir associer à plusieurs centaines de schémas d'arbres élémentaires ; pouvoir réduire ce nombre à quelques unités permet d'assurer un temps d'analyse raisonnable.

Dans [Srinivas97a] sont reportées les expérimentations de différents modèles probabilistes pour le *supertagging*, en particulier :

- un modèle *trigramme* classique en étiquetage automatique qui atteint une correction

par mot de 92,4% sur 47 000 mots du corpus *Wall Street Journal* avec un apprentissage initial sur 1 million de mots [Srinivas97b] ;

- un modèle de dépendance exploitant les informations de sous-catégorisation encodées dans les *supertags*, l'analyseur fonctionnant alors en plusieurs passes chargées, toujours sur la base d'apprentissage stochastique, de localiser les dépendances sémantiques. Ce modèle, bien que motivé linguistiquement, donne de moins bons résultats d'étiquetage qu'un simple trigramme, mais offre des résultats plus intéressants car très proches d'analyses syntaxiques partielles et d'analyses robustes.

Notons qu'en tant que modèle de langage, le *supertagging* atteint un score de 93,8% d'étiquettes correctes sur le corpus de dialogue homme-machine ATIS. La perplexité obtenue sur le corpus du Wall Street Journal est de 6,23 contre 7.61 pour un modèle classique [Srinivas97a].

Les performances du *supertagging* sont donc encourageantes étant donné la complexité de la tâche, qui peut atteindre plusieurs centaines de *supertags* à désambiguïser, et rapides (pouvant atteindre plusieurs dizaine de mots à la seconde). Les applications proposées par Srinivas sont d'ailleurs nombreuses : de l'analyse robuste à l'extraction d'informations, en passant par la simplification de textes, la désambiguïsation sémantique de mots, la traduction et l'identification de groupes nominaux. Notons cependant que les performances ont atteint un certain plafonnement, ne s'améliorant plus avec l'augmentation de la taille du corpus, et que les nouveaux modèles de *supertagging* expérimentés dans [Chen et al.99] n'ont au mieux amélioré les résultats que de deux dixièmes.

2.3.3 Probabilité *inside* et *outside* pour l'analyse de TAG

Dans [Halber98b] est proposé un modèle d'analyse stochastique de LTAG par *chart* intégrant le modèle stochastique de [Schabes92] et le *supertagging*. En se basant sur les travaux de [Goodman98], l'idée est de voir l'analyse stochastique comme modélisant les probabilités *inside*, c'est-à-dire la probabilité de la dérivation en cours vue comme le produit des probabilités des sous-dérivations la constituant, et le *supertagging* comme modélisant les probabilités *outside*, c'est-à-dire les probabilités qu'une dérivation partielle conduise à une dérivation globale étant données les autres dérivations partielles environnantes.

Dans ce séduisant modèle probabiliste, l'analyse proposée est ascendante mais n'exploite pas les co-ancres et donc l'ensemble du domaine de localité. Il est en effet problématique de définir sur quels critères effectuer la combinaison des probabilités entre deux *items* correspondant à l'extension de deux ancres d'un même arbre. L'implantation effective du modèle et sa validation, suite à un entraînement sur un volume de données suffisant, permettra de déterminer si l'estimation des probabilités n'est pas trop coûteuse en temps de calcul.

2.4 Simplification du formalisme TAG

Les premières implantations du formalisme LTAG ont conduit à des résultats trop lents pour des applications soumises à des contraintes de fonctionnement en temps-réel, comme les systèmes de dialogue. Une analyse avec le système XTAG, bien que reposant sur un analyseur prédictif de type Earley, peut prendre facilement plusieurs minutes. Ces contraintes d'implantation ont justifié une approche fondée sur la dégradation du pouvoir d'expression pour rendre le formalisme équivalent à une grammaire hors contexte.

2.4.1 Les TIG

Dans la section 1.1.1 de cette deuxième partie du mémoire, nous avons introduit les Grammaires d'Arbres Inserés ou TIG [Schabes et al.95a], simplification du formalisme TAG permettant une équivalence faible et forte avec les grammaires CFG tout en préservant les propriétés linguistiques et informatiques intéressantes du formalisme initial.

Cette simplification part du constat suivant : la grammaire XTAG de l'anglais ne contient en fait aucun arbre auxiliaire englobant. Cette remarque peut également s'appliquer à la grammaire TAG du français [Candito99] où on privilégie un contrôle plus souple des contraintes par un mécanisme de trait.

La question fondamentale derrière cette constatation est la suivante : qu'apporte la puissance formelle supplémentaire des Grammaires LTAG permise par l'adjonction englobante ? En quoi le coût d'analyse supplémentaire est-il justifié s'il n'apporte pas plus de facilité pour exprimer des phénomènes complexes ? Si la puissance formelle supplémentaire permise par une grammaire DTG s'explique par la prise en compte de nombreux phénomènes comme le *scrambling* et la localisation des dépendances syntaxiques aussi bien que sémantiques, elle ne semble pas se justifier dans une grammaire LTAG. Les propriétés intéressantes du formalisme que sont le domaine de localité étendu, la lexicalisation et la localisation des dépendances sémantiques sont également partagées par une grammaire TIG. De façon plus précise, notre travail devra avant tout prendre en compte cette constatation. Nous concevons en particulier l'adjonction englobante plus comme un problème formel que comme une préoccupation pratique réelle. De plus, le coût d'analyse supérieur pour une grammaire TIG ou TAG par rapport à une grammaire hors contexte, que l'on observe par exemple avec XTAG, provient bien de la lexicalisation et du domaine de localité étendu qui entraîne une multiplication massive des sous-structures communes.

2.4.2 Les TFG

Le formalisme des grammaires d'Arbres Furquants (TFG) va nettement plus loin dans une simplification des TAG, en étant clairement motivé par des questions de perfomances informatiques. Inspiré par l'opération de furcation d'arbres introduite par [Smedt et al.90], la composition d'arbres n'entraîne pas ici l'introduction d'un niveau syntagmatique supplémentaire à la différence de l'adjonction des TAG. Les TFG ont été utilisées dans [Roussel et al.98a] avec comme objectif premier de permettre une intégration avec la sémantique représentée sous forme de traits décorant les arbres élémentaires TFG.

Les principales simplifications sont les suivantes :

- une unique structure de traits par arbre élémentaire et par nœud à substituer (avec la possibilité dans ce cas de disjoindre des structures de traits). Ceci simplifie de manière notable les unifications et permet de filtrer les arguments possibles d'une opération par les traits. L'unification en deux temps des TAG pose en effet un problème concernant la prédictivité des traits dans un contexte d'analyse d'hypothèses de reconnaissance de la parole ;
- un abandon du principe prédicat-argument, ce qui permet de limiter le nombre d'arbres par rapport à une grammaire LTAG classique. Les sous-catégorisations sont prises en compte par la sémantique associée qui est fortement lexicalisée.

L'abandon du principe de coocurrence prédicat-argument qui repose sur un système de traits nous paraît difficile à justifier, puisque c'est là la propriété la plus intéressante des formalismes d'arbres lexicalisés. Certes la combinatoire au niveau structurel est diminuée,

mais la prise en compte des structures prédicats/arguments par des arbres décorés de traits est bien plus coûteuse car il n'y a pas de tabulation possible [Carroll et al.99a].

L'intérêt méthodologique, les justifications et les phénomènes qu'il n'est plus possible de prendre en compte au niveau structurel sont présentés dans [Roussel et al.99b].

A la différence de cette approche, notre hypothèse est que la complexité des traitements peut être paramétrée vis-à-vis de la complexité des données à analyser, sans pré-supposition dans la conception des outils. Autrement dit, si une tâche justifie l'utilisation d'une simplification de type TFG avec un seul ensemble de traits par nœud, la complexité du traitement avec des outils dédiés de manière générale aux TAG doit s'ajuster aux données. Ceci suppose bien entendu des choix algorithmiques et d'implantation particuliers.

Nous pouvons illustrer cette idée avec l'implantation classique d'un algorithme d'unification sur des DAG générales. La complexité d'analyse est alors exponentielle par rapport à la taille des termes mis en jeu. Si on suppose que les traits sont atomiques, la complexité passe alors à n^2 sans que l'algorithme doive être modifié, n étant la taille maximale des termes à unifier. Nous allons maintenant nous intéresser de façon plus détaillée aux techniques d'analyse existantes pour les grammaires LTAG et à leur adéquation aux contraintes et aux objectifs fixés pour le traitement de l'oral.

Chapitre 3

Techniques d'analyse pour les LTAG

3.1 Problématique

La mise en œuvre d'analyses polynomiales pour le formalisme LTAG suppose d'employer des méthodes capables de factoriser les parties analysées relativement à leur position sur la chaîne d'entrée. Dans la mesure où l'analyse que nous souhaitons mettre en œuvre est non déterministe, l'espace de recherche d'un algorithme exponentiel sera trop important pour des applications pratiques impliquant un nombre de mots élevé et une grammaire réaliste.

Une analyse polynomiale n'est cependant pas en soi une garantie de performance. En effet, on peut constater qu'un algorithme polynomial sur une grammaire très redondante s'avère inutilisable. Par exemple une simple mise sous forme normale de Chomsky d'une grammaire hors contexte (chaque règle de réécriture ne contient en partie droite qu'au plus deux symboles non-terminaux ou terminaux) permet de factoriser les sous-parties communes de cette grammaire et de rendre plus efficace une telle analyse.

Notons que la limite de complexité au pire des cas en $\mathcal{O}(n^6)$ des algorithmes existants pour les LTAG n'est pas une limite théorique. En effet, [Satta94] a montré qu'il était possible d'analyser une grammaire LTAG suivant des méthodes asymptotiquement plus rapides en temps que $\mathcal{O}(n^6)$. Cependant ces méthodes sont en pratique inutilisables à cause de constantes beaucoup trop importantes. En considérant certaines sous-classes plus ou moins linguistiquement motivées de TAG, il est également possible de diminuer la complexité au pire des cas, par exemple en $\mathcal{O}(n^5)$ dans [Satta et al.98]. Cependant nous considérons qu'en pratique, lorsqu'on analyse la langue naturelle, diminuer la complexité moyenne d'analyse et améliorer la robustesse des résultats partiels est plus important que la considération d'un cas de complexité au pire des cas. Cette complexité au pire des cas, d'une part, reste polynomiale et, d'autre part, les conditions d'apparition d'une telle complexité ne sont en pratique jamais réunies (voir section 2.4.1).

Les algorithmes d'analyse existant pour le formalisme LTAG sont généralement issus d'algorithmes pour grammaires hors contexte. Le compromis entre analyses prédictives ou non et analyses robustes présenté dans la première partie de ce mémoire se retrouve avec la même acuité pour les grammaires arborescentes.

Un premier choix fondamental se pose au niveau de la conception d'un analyseur pour les grammaires LTAG : faut-il travailler sur des précompilations des actions de l'analyse (table LR, automate) ou sur des résultats temporaires générés dynamiquement (*chart*) ?

3.1.1 Les algorithmes de type LR

Ces algorithmes ont été conçus pour les grammaires hors contexte en vue de fournir une unique solution d'analyse. Ils reposent sur la compilation d'un automate fini guidant les actions à effectuer lors de l'analyse. Cet automate est en pratique représenté à l'aide d'un table stockant les actions à effectuer. Ces algorithmes peuvent être adaptés à l'analyse de TAG lexicalisées à l'aide d'automates plus puissants [Schabes et al.90b] [Sarkar96] [Kinyon97] [Nederhof98] et à l'analyse non déterministe à l'aide de piles structurées en graphe [Tomita87]. L'idée est de précompiler dans une table les actions de l'analyseur. Il n'y a donc pas de structures temporaires d'analyse indexées sur lesquelles faire facilement un retour en arrière. Un second problème pratique est la taille des tables compilées qui peut atteindre plusieurs dizaines de MégaOctets. Par exemple avec l'algorithme prédictif de [Nederhof98], qui inclu un partage des sous-structures communes, on atteint plus de 40 Mo pour 600 schèmes de XTAG. Dans le cas d'un algorithme ascendant par îlots suivant l'approche proposée par [Satta et al.91], la table atteindrait des dimensions encore plus considérables. Nederhof conclut sur la nécessité d'une interprétation tabulaire de son algorithme en vue d'une mise en œuvre pratique à l'aide d'un *chart*.

3.1.2 Les algorithmes tabulaires par *chart*

L'idée des techniques tabulaires est de sauvegarder des structures temporaires de travail dans une table. Pour l'analyse syntaxique, ces structures temporaires correspondent à des sous-dérivations partageables. Ce principe permet de prendre en compte des répétitions de calculs et d'éviter des boucles. Stockés dans une table, les résultats d'un calcul peuvent être réutilisés de façon optimale dans la mesure où les dérivations sont indépendantes du contexte. Fort heureusement, ce dernier point est une propriété vérifiée par les grammaires LTAG, ceci même pour la règle d'adjonction englobante. [VS87] a montré qu'une grammaire LTAG pouvait être vue comme une grammaire hors contexte d'arbres. C'est cette propriété qui permet notamment la réalisation d'un arbre de dérivation où chaque nœud correspond à un arbre élémentaire.

Les techniques tabulaires sont particulièrement adaptées à la mise en œuvre d'algorithmes non déterministes où il est nécessaire de disposer facilement des différentes alternatives et des résultats partiels. Un exemple d'application de techniques tabulaires est la *memoization* en prolog et les *magic sets* pour les bases de données déductives.

En informatique linguistique, les techniques tabulaires permettent de répondre aux problèmes posés par l'ambiguïté inhérente à la langue, tout en permettant des analyses polynomiales, par exemple pour les grammaires hors contexte. Ces techniques prennent le plus souvent la forme d'un *chart*, l'algorithme CKY (Cooke Kasami Younger) en étant l'exemple le plus simple.

3.2 Algorithmes ascendants

3.2.1 Algorithme de type CKY

Principes

Le premier algorithme destiné à l'analyse de TAG, historiquement parlant, est une adaptation de l'algorithme CKY [VS87]. Cette approche correspond à un processus d'analyse purement ascendant.

Considérons la chaîne d'entrée à analyser suivante w, de longueur n :

$$w = a_1 a_2 a_3 ... a_n$$

Nous appellons *indice* un entier entre 0 et n, noté ci-dessous entre crochets, identifiant les segments de w :

$$[0]a_1[1]a_2[2]a_3...[n-1]a_n[n]$$

Formellement, une Grammaire d'Arbres Adjoints peut être spécifiée par un quintuplet $G = (\Sigma, NT, I, A, S)$, où Σ est l'ensemble des symboles terminaux, NT l'ensemble disjoint de Σ des non-terminaux incluant le symbole de départ (ou axiome) S, et où I et A sont deux ensembles finis d'arbres appellés respectivement arbres initiaux et arbres auxiliaires. On considère la définition standard d'une grammaire TAG entièrement lexicalisée, donc dans laquelle chaque arbre élémentaire contient au moins un nœud feuille, appellé *ancre*, correspondant à un mot. Pour les résultats de complexité, nous notons N le nombre maximal de nœuds d'un arbre élémentaire et G la taille de l'ensemble $I \cup A$. Le nœud racine d'un arbre γ sera noté $racine(\gamma)$. Enfin, nous notons $spine(\gamma)$ l'ensemble des nœuds appartenant à l'épine dorsale d'un arbre auxiliaire γ, c'est-à-dire l'ensemble des nœuds appartenant à l'unique chemin entre le nœud racine et le nœud pied de γ.

Les algorithmes d'analyse de LTAG inspirés des algorithmes existant pour les CFG utilisent souvent une marque (un point, *dot* en anglais) pour indiquer l'avancement de l'analyse dans une règle de réécriture. Cette idée est reprise ici, indiquant cette fois une position dans un arbre élémentaire. On parle alors *d'arbre pointé* (*doted tree* en anglais). L'évolution du *point* indique le parcours d'analyse.

L'évolution du *point* dans un arbre élémentaire pour l'algorithme CKY adapté aux LTAG est illustrée figure 3.1. Tous les arbres élémentaires sont reconnus de manière ascendante après avoir été mis, au préalable, sous une forme normale de Chomsky, c'est-à-dire sous la forme d'un arbre binaire.

Le point indique si l'adjonction (ou la non adjonction) au nœud correspondant a été prise en compte (*point* en position haute) ou non (*point* en position basse).

Il est également possible de considérer chaque arbre élémentaire γ comme le résultat d'un ensemble de productions hors contextes $\mathcal{P}(\gamma)$: un nœud N^γ de γ et ses fils $N_1^\gamma,...,N_p^\gamma$ sont alors représentés par une production $N^\gamma \rightarrow N_1^\gamma,...,N_p^\gamma$. La position du *point* dans un arbre élémentaire γ peut être indiquée ici par une position du *point* dans une des productions de $\mathcal{P}(\gamma)$. Cette façon de représenter un arbre pointé, introduite par [Nederhof97] est intéressante car elle permet d'indiquer un avancement de l'analyse par rapport aux frères et aux fils d'un nœud donné [Nederhof98]. C'est cette représentation que nous emploierons dans la suite.

L'algorithme manipule des états d'avancement de l'analyse nommés *item*, il s'agit d'un six-uplet du type :

$$(N^\gamma, i, l, j, k, adj)$$

Cet *item* signifie que le sous-arbre dominé par N de l'arbre élémentaire γ contenant N a été entièrement reconnu (avec toutes les adjonctions et substitutions possibles) et recouvre la chaîne d'entrée entre les indices i et l. j et k donnent la position du sous-arbre dominé par l'éventuel nœud pied de cet arbre élémentaire. Notons que par rapport à l'analyse d'une grammaire hors contexte, il est nécessaire de rajouter ces deux derniers indices afin de

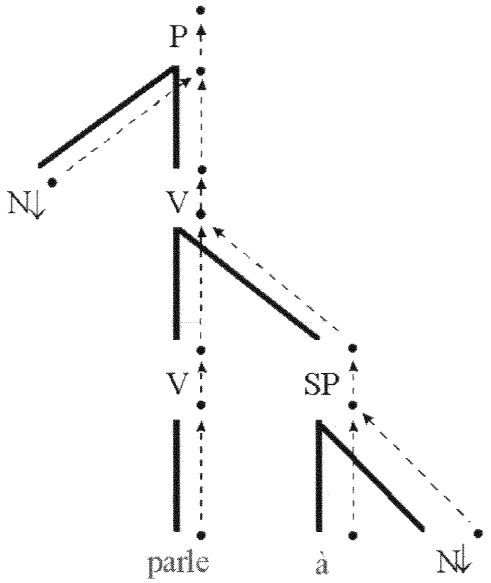


FIG. 3.1 – *Évolution du point lors de l'analyse d'un arbre élémentaire avec l'algorithme de type CKY [Shieber et al.95].*

prendre en compte la possibilité, pour les LTAG, d'avoir deux sous-chaînes non contiguës, ce qui est là l'origine de toutes les difficultés d'analyse avec une telle grammaire. S'il ne s'agit pas d'un arbre auxiliaire, j et k sont notés $-$. Le booléen *adj* indique si une adjonction a eu lieu au nœud N^{γ} ($vrai$), ou non ($faux$). Ce booléen permet de prendre en compte la contrainte d'une seule adjonction par nœud [VS87].

L'algorithme CKY passe le plus souvent par la création d'un *item* pour une position du point dans un arbre élémentaire et par segment dominé de la chaîne. Ces *items* sont introduits par un ensemble de règles qui simulent ainsi, au fur et à mesure de l'introduction de nouveaux *items*, la marche du *point* dans les arbres élémentaires.

Spécifications des algorithmes d'analyse

La spécification des algorithmes d'analyse que nous proposons dans ce mémoire de thèse se fonde sur les schémas d'analyse de [Sikkel97] et les systèmes de déduction grammaticale de [Shieber et al.95]. Ces deux approches sont similaires et correspondent à des descriptions de haut niveau d'algorithmes d'analyse, tout à fait adaptées au formalisme des LTAG [Alonso et al.99]. Un système d'analyse pour une grammaire G et une chaîne donnée est un triplet $(\mathcal{I},\mathcal{H},\mathcal{D})$. $\mathcal{I}$ est un ensemble d'*items* qui représente la progression de l'analyse (habituellement cet ensemble est stocké dans un *chart*). $\mathcal{H}$ est un ensemble initial d'*items*, les hypothèses, encodant la chaîne d'entrée à analyser. Enfin, $\mathcal{D}$ est un ensemble d'étapes déductives exprimées sous la forme de règles du type :

$$\frac{item_1 \quad \ldots \quad item_n}{item_e} \quad (\text{ conditions })$$

Le processus d'analyse peut être vu comme l'exécution d'un ensemble de règles d'inférence utilisées pour introduire de nouveaux *items*. Ces règles ont la signification suivante :

si les $(item_i)_{1 \leq i \leq n}$ sont présents dans $\mathcal{I}$ et s'ils satisfont les conditions exprimées, alors il convient d'ajouter $item_e$ dans $\mathcal{I}$.

Un schéma d'analyse est équivalent au système de déduction grammatical de [Shieber et al.95], les *items* y étant appellés formules schématiques (*formula schemata*), les étapes de déduction se nommant règles d'inférence, les hypothèses axiomes et les *items* finaux formules de but (*goal formula*). En plus d'une grande lisibilité, cette formalisation de description permet de voir certains algorithmes comme la généralisation d'autres en se basant sur un petit nombre de règles de transformation.

On peut considérer deux principales étapes lors la mise en œuvre d'un algorithme : une étape d'initialisation et une étape de production d'*items* suivant les règles d'inférence. Nous allons présenter ces deux étapes suivant ce type de description pour l'algorithme CKY adapté aux LTAG.

Initialisation des items

L'initialisation pour l'algorithme CKY adapté consiste en la création des *items* associés aux positions des nœuds feuilles des arbres élémentaires. L'algorithme étant ascendant, on suppose que l'on dispose de l'ensemble de la chaîne à analyser et des positions possibles des différents mots. L'initialisation a lieu ainsi :

- pour les ancres : il suffit de créer les *items* $(N^\gamma, i, l, -, -, faux)$, où $N^\gamma \rightarrow \Diamond$, $\Diamond$ étant une marque lexicale dans l'arbre élémentaire γ, et où i et l sont les indices associés au mot ancrant l'arbre ;
- pour les nœuds pieds : on crée des *items* $(N \star^\gamma, i, l, i, l, faux)$ pour chaque couple $(i,l)_{i \in [0,n], l \in [0,n], i<l}$;
- les nœuds de substitution sont initialisés au moment de la reconnaissance de chaque nœud racine. Dans les descriptions habituelles de l'algorithme CKY pour les LTAG, la substitution n'est pas prise en compte, le fonctionnement étant identique à celui de l'opération de réécriture des grammaires hors contextes ;
- si la grammaire emploie des catégories vides (notées ε), on crée les items $(N^\gamma, i, i, -, -, faux)$, où $N^\gamma \rightarrow \varepsilon$.

De façon similaire aux étapes de déduction, ces initialisations peuvent être exprimées sous forme de règles d'inférence, que nous illustrons, table 3.1, à l'aide de correspondances avec des arbres élémentaires stylisés.

Remarquons que la phase d'initialisation des ancres n'est pas triviale, puisque, pour chaque arbre élémentaire, il est nécessaire d'initialiser chaque possibilité d'occurrence dans l'énoncé de l'ancre principale et de ses co-ancres. Cette étape est de complexité au pire des cas en $\mathcal{O}(n^2)$ (voir par exemple [Issac97] pour un calcul de cette complexité).

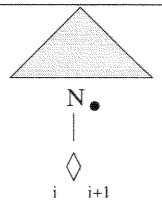
Production des items

Les tableaux 3.3 et 3.2 illustrent le système de règles d'inférence correspondant à l'algorithme d'analyse CKY pour les LTAG [Alonso et al.99]. La plupart de ces règles, celles du tableau 3.3, sont en charge du parcours ascendant des arbres élémentaires. Dans le cas d'arbres auxiliaires, c'est-à-dire pour la montée binaire du point cas 2 et 3, ces règles rendent compte de la propagation du segment de chaîne reconnu par le nœud pied.

L'initialisation des nœuds pieds à tous les segments possibles de l'énoncé conduit à un nombre important d'hypothèses d'analyse pour les arbres auxiliaires. Bien entendu un

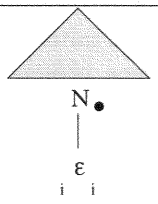
Initialisation des ancres :

$$\frac{}{(N^\gamma,i,i+1,-,-,faux)}\ (N^\gamma \rightarrow \Diamond)$$



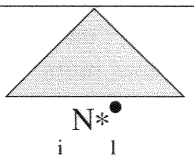
Initialisation des catégories vides :

$$\frac{}{(N^\gamma,i,i,-,-,faux)}\ (\ N^\gamma \rightarrow \varepsilon)$$



Initialisation des nœuds pieds :

$$\frac{}{(N\star^\gamma,i,l,i,l,faux)}\ (\ \forall(i,l),i,l \in [0,n],i<l)$$



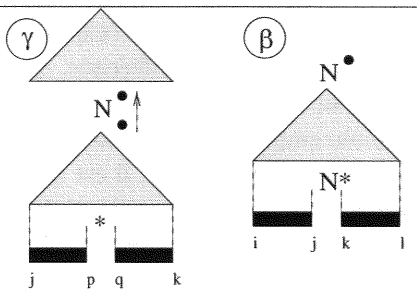
TAB. 3.1 – *Règles d'inférences pour l'initialisation des items dans l'algorithme CKY.*

très grand nombre de ces hypothèses ne prendront part à aucune dérivation. Seules celles compatibles avec des nœuds sites d'adjonction seront utilisées par la règle d'adjonction illustrée table 3.2.

Adjonction :

$$\frac{(N^\beta,i,l,j,k,adj) \qquad (N^\gamma,j,k,p,q,false)}{(N^\gamma,i,l,p,q,vrai)}$$

tel que $\beta \in A, N^\beta = racine(\beta)$

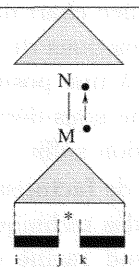


TAB. 3.2 – *Règle d'inférence pour l'adjonction dans l'algorithme CKY.*

L'analyse se termine lorsque plus aucune opération ne peut s'appliquer. L'espace de recherche a été entièrement parcouru et l'ensemble des dérivations évaluées. Les solutions complètes sont données par l'ensemble des *items* de la forme $(N^\alpha,0,n,-,-,adj)$, où $N^\alpha = racine(\alpha)$ et $alpha \in I$.

Montée unaire du point :

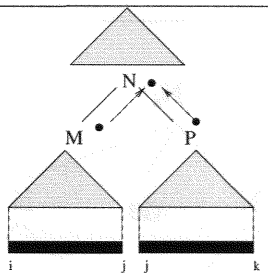
$$\frac{(M^\gamma,i,l,j,k,adj)}{(N^\gamma,i,l,j,k,faux)}\ (N^\gamma \rightarrow M^\gamma \in \mathcal{P}(\gamma))$$



Montée binaire du point cas 1 :

$$\frac{(M^\gamma,i,j,-,-,adj) \qquad (P^\gamma,j,k,-,-,adj)}{(N_\bullet,i,k,-,-,faux)}$$

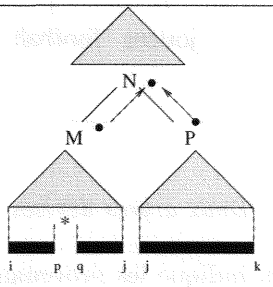
tel que $(N^\gamma \rightarrow M^\gamma P^\gamma \in \mathcal{P}(\gamma))$



Montée binaire du point cas 2 :

$$\frac{(M^\gamma,i,j,p,q,adj) \qquad (P^\gamma,j,k,-,-,adj)}{(N^\gamma,i,k,p,q,faux)}$$

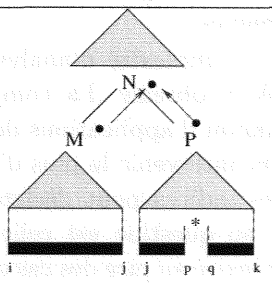
tel que $(N^\gamma \rightarrow M^\gamma P^\gamma \in \mathcal{P}(\gamma), M^\gamma \in spine(\gamma)$



Montée binaire du point cas 3 :

$$\frac{(M^\gamma,i,j,-,-,adj) \qquad (P^\gamma,j,k,p,q,adj)}{(N^\gamma,i,k,p,q,faux)}$$

tel que $(N^\gamma \rightarrow M^\gamma P^\gamma \in \mathcal{P}(\gamma)), P^\gamma \in spine(\gamma)$



TAB. 3.3 – *Règles d'inférence pour le parcours des arbres élémentaires dans l'algorithme CKY.*

Analyse à l'aide d'un *chart*

Nous avons indiqué que l'ensemble $\mathcal{I}$ des *items* se réalisait, en pratique, avec un *chart*. Il nous faut expliquer plus précisément ce que nous appellons ici *chart* et l'intérêt de

son utilisation. Chaque *item* du type (N^{γ},i,l,j,k,adj) est rangé dans une table à quatre dimensions appellée *chart* dans la position correspondant aux indices (i,l,j,k). Chacune de ces différentes positions du *chart* est susceptible d'accueillir une liste d'*items*. Si on doit ajouter un *item* à une position déjà occupée par un *item* identique, celá signifie qu'on a obtenu une même sous-dérivation de deux façons différentes. Dans ce cas de figure, une seule représentation suffit.

Avec ce type de factorisation, l'algorithme de type CKY que nous venons de présenter illustre l'intérêt des techniques tabulaires. Par exemple l'item suivant : (N,0,5, $-$, $-$,$vrai$), où N est le nœud racine de l'arbre initial associé au nom *joueur*, factorise en fait la reconnaissance des deux arbres dérivés donnés par la figure 3.2 pour l'énoncé ambigu : *le joueur de football américain*.

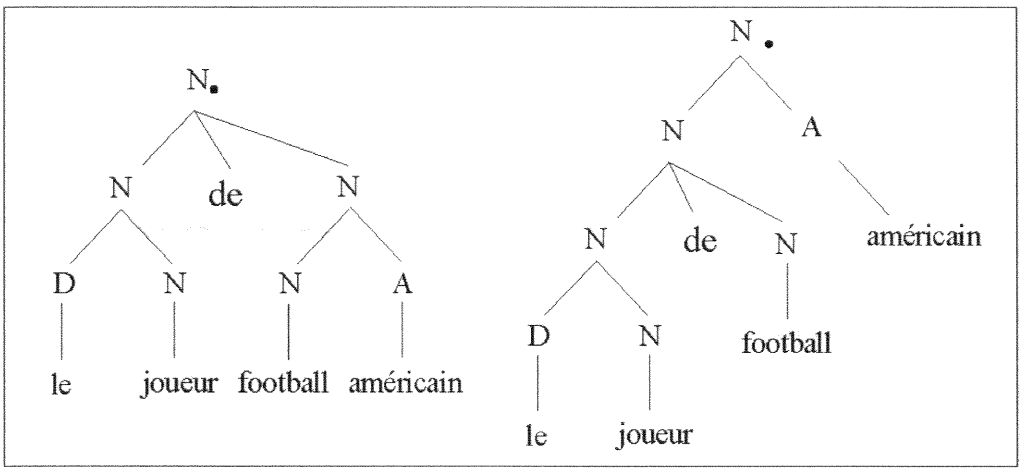


FIG. 3.2 – *Deux arbres dérivés représentés par le même item d'analyse* (N,0,5, $-$, $-$,$vrai$).

On indique les provenances de chaque *item* par les couples ou les éventuels singletons d'*items* l'ayant produit et par la règle d'inférence employée. Le *chart* devient un graphe orienté acyclique tabulant tous les *items* partagés, c'est-à-dire toutes les sous-dérivations communes.

La complexité d'analyse, dans le cadre adopté de spécification d'algorithme, est très simple à obtenir. La complexité temporelle au pire des cas est donnée par le nombre maximum d'applications des règles. Ce nombre est lui-même majoré par la règle d'inférence faisant intervenir le plus d'indices libres. Dans le pire des cas, chaque distribution possible de ces indices peut déclencher cette règle. Pour l'algorithme de type CKY présenté, la règle en question est celle de l'adjonction. Elle met en œuvre 6 indices libres, soit une complexité au pire des cas en $\mathcal{O}(n^6)$ concernant la longueur de la chaîne d'entrée. Cette règle combine deux arbres élémentaires, donnant au pire des cas une complexité relativement à la taille de la grammaire en $\mathcal{O}(G^2)$. Enfin, chacun des 2 *items* contient une position sur un arbre élémentaire, mais une seule est libre, l'autre étant invariablement la racine de l'arbre auxiliaire adjoint, donnant une complexité au pire des cas en $\mathcal{O}(N)$. La complexité spatiale est donnée par le nombre d'*items* dans le *chart*. Dans le pire des cas, il s'agit de toutes les distributions possibles pour le six-uplet correspondant à la définition d'un *item*, soit $\mathcal{O}(n^4)$ car présentant quatre positions libres d'indice, $\mathcal{O}(G)$ chaque arbre élémentaire pouvant être lié à un *item* et $\mathcal{O}(N)$ pour toutes les positions possibles sur ces arbres élémentaires.

On associe classiquement à chaque *item* un *historique* chargé d'interdire l'application d'opérations ayant déjà été effectuées. C'est par cette méthode que l'on évite des récursions infinies. L'idée est d'associer à chaque *item* la liste des opérations et arguments ayant déjà été utilisés avec cet *item*. Au cours de l'analyse, étant donné une opération et un premier *item* opérande, on évaluera un *item* argument que si ces deux *items* n'apparaissent pas dans leurs historiques respectifs.

Enfin, on emploie couramment avec un *chart* une structure de données, l'*agenda*, chargée d'indiquer suivant quel ordre les *items* doivent être examinés pour l'évaluation des opérations. En pratique, il s'agit d'une simple liste d'*items*. C'est par l'intermédiaire de l'agenda qu'on peut appliquer certaines heuristiques comme privilégier les derniers *items* produits (stratégie LIFO, *last in, first out*), les *items* les plus étendus ou ne considérer que les *items* couvrant un préfixe donné (stratégie de sélection Earley). En pratique, un agenda permet d'améliorer de façon intéressante les temps moyens d'analyse. Dans des espaces de recherche pouvant être très vastes, un agenda maintenu suivant de bonnes heuristiques permet d'accéder très vite aux *items* fortement susceptibles de participer à des dérivations. Notons cependant que l'efficacité d'un agenda dépend de l'algorithme d'analyse considéré et, bien évidemment, du choix des heuristiques employées. Sans agenda, l'ordre dans lequel sont évalués les *items* correspond en fait à un simple parcours du *chart*, par exemple de gauche à droite suivant les indices respectifs.

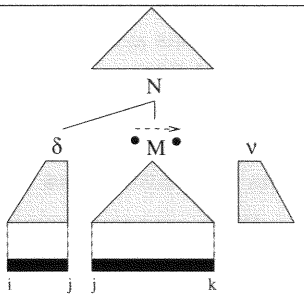
3.2.2 CKY généralisé et CKY prédictif

L'adaptation de l'algorithme CKY pour l'analyse de LTAG offre de très mauvaises performances. Pour Krishnamurti Vijay-Shanker, cet algorithme était plus un résultat théorique qu'un algorithme effectivement utilisable dans une application [VS87]. Le principal problème est l'initialisation de l'ensemble des nœuds pieds de la grammaire sur tous les segments possibles de la chaîne d'entrée, c'est-à-dire n^2 possibilités. La complexité en moyenne résultante est très élevée car, présentant toujours des nœuds pieds compatibles, des adjonctions seront évaluées en très grand nombre au cours de la reconnaissance des arbres initiaux. C'est seulement plus loin dans l'analyse que ces hypothèses tomberont car incohérentes avec les possibilités de combinaisons avec les autres *items*.

Une première amélioration de cet algorithme a consisté à éliminer la contrainte d'une mise sous forme normale. On parle alors d'algorithme de type CKY généralisé. L'idée est simplement de reconnaître successivement tous les nœuds d'un même niveau d'arborescence de gauche à droite. Un point au niveau de la règle de production associée au niveau courant de l'arborescence (voir section 3.2.1) indique la position courante de l'analyse. Cette notation rend inutile le booléen *adj*: elle indique que les non-terminaux à gauche du point ont déjà été évalués pour une adjonction, la partie droite restant à reconnaître. Le tableau 3.4 illustre cette notation avec la remontée générale du point (valide quel que soit le nombre de fils par nœud). Le sous-arbre dominé par le nœud M^γ ayant entièrement été reconnu entre j et k, on peut avancer le point d'un cran sur la droite dans la règle de réécriture où M^γ figure, c'est-à-dire dans $N^\gamma \rightarrow \delta \bullet M^\gamma \nu$. Dans ce tableau δ, ν et v désignent une séquence de terminaux ou de non terminaux pouvant être nulle.

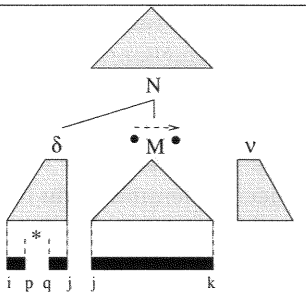
Montée générale du point cas 1 :

$$\frac{(N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, -, -) \qquad (M^\gamma \rightarrow v \bullet, j, k, -, -)}{(N^\gamma, i, k, -, -)}$$



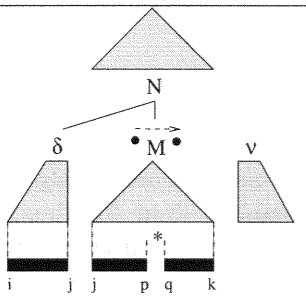
Montée générale du point cas 2 :

$$\frac{(N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, p, q) \qquad (M^\gamma \rightarrow v \bullet, j, k, -, -)}{(N^\gamma, i, k, p, q)}$$



Montée générale du point cas 3 :

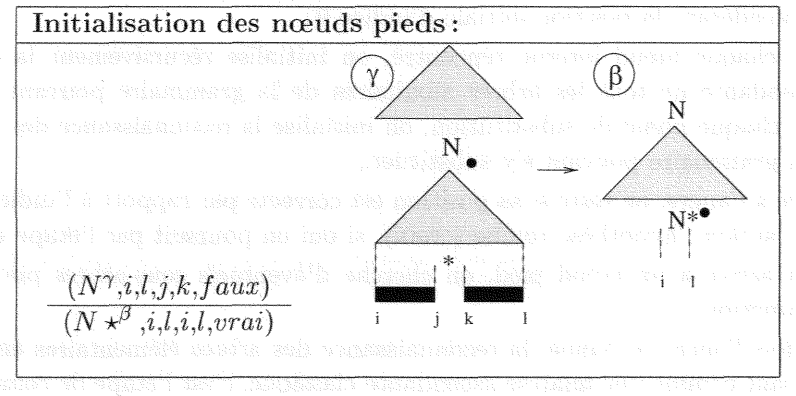
$$\frac{(N^\gamma \rightarrow \delta \bullet M^\gamma \nu, i, j, -, -) \qquad (M^\gamma \rightarrow v \bullet, j, k, p, q)}{(N^\gamma, i, k, p, q)}$$



TAB. 3.4 – *Règles d'inférence pour la montée du point dans l'algorithme CKY généralisé.*

Une seconde amélioration de l'algorithme de type CKY constitue en l'injection de prédiction dans l'initialisation des nœuds pieds. Il en résulte un algorithme nettement plus dirigé par le but. Par prédiction il faut comprendre ici l'application d'un filtre dynamique dont le but est de diminuer la taille de l'espace de recherche. L'idée est de ne considérer à un moment donné de l'analyse que les *items* utiles au succès du processus d'analyse. En effet, la plupart des initialisations de nœuds pieds, réalisées avec la version purement ascendante de l'algorithme de type CKY que nous avons présenté, sont inutiles. L'application de prédiction consiste ici à ne considérer que les *items* compatibles avec une adjonction. En pratique, il s'agit de retarder l'initialisation des nœuds pieds, cette initialisation étant une conséquence de la reconnaissance d'un nœud site possible de l'adjonction, comme illustré par le tableau 3.5. Il en résulte une complexité moyenne d'analyse nettement plus fible et donc une efficacité accrue. Néanmoins l'initialisation des nœuds pieds reste encore très

systématique, s'appliquant à tous les nœuds pieds de la grammaire indépendamment des positions des ancres des arbres auxiliaires.



TAB. 3.5 – *Règle d'inférence pour l'initialisation des nœuds pieds dans l'algorithme CKY prédictif.*

3.3 Algorithmes prédictifs et descendants

3.3.1 Algorithme de type Earley

L'algorithme d'Earley a été adapté au LTAG dès 1988 [Schabes et al.88], présentant une complexité au pire des cas en $\mathcal{O}(n^6)$ [Lang90] [Schabes94]. Le parcours associé à cet algorithme d'analyse est illustré figure 3.3. L'idée est d'incorporer à l'analyse des prédictions descendantes afin de réduire l'espace de recherche uniquement aux *items* pouvant être utiles compte tenu d'un contexte gauche reconnu.

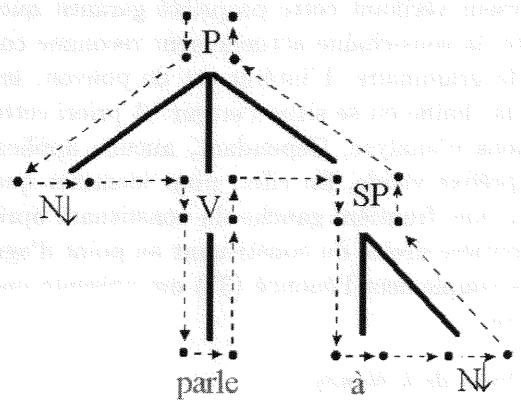


FIG. 3.3 – *Évolution du point lors de l'analyse d'un arbre élémentaire avec l'algorithme de type Earley.*

La figure 3.4 illustre le fonctionnement de l'algorithme. Ce dernier met en œuvre trois étapes principales : une étape de prédiction (*prediction*), une de lecture de la chaîne d'entrée

(*scanner*) et une de complétion (*complete*). Ces étapes sont mises en œuvre en parallèle de la façon suivante :

1. on débute la reconnaissance descendante de tous les arbres ayant pour racine l'axiome en considérant la position initiale d'indice 0 ;
2. pour chaque nœud interne rencontré, on initialise récursivement la reconnaissance descendante de tous les arbres auxiliaires de la grammaire pouvant s'y adjoindre : pour chaque nœud de substitution, on initialise la reconnaissance des arbres initiaux de la grammaire pouvant s'y substituer ;
3. arrivé à l'ancre, on teste si sa position est correcte par rapport à l'indice prédit (*scanner*) : si non l'hypothèse tombe (croix), si oui on poursuit par l'étape de complétion ;
4. si on arrive à un nœud pied, on cherche d'éventuels sous-arbres pouvant accueillir l'adjonction ;
5. une fois l'ancre reconnue, la reconnaissance des arbres élémentaires encore valides se poursuit comme une analyse ascendante classique, c'est l'étape de complétion qui elle aussi fera tomber certaines hypothèses.

Par son utilisation intensive des prédictions, l'algorithme d'Earley est un des plus efficaces qui soient. De nombreux arbres élémentaires qui auraient l'objet d'une reconnaissance ascendante et de calcul coûteux par exemple par l'algorithme de type CKY, ne sont même pas envisagés ici car incohérents avec les préfixes déjà reconnus. Par contre, le prix de cette stratégie entièrement dirigée par le but se répercute sur la robustesse lorsque l'énoncé n'est pas grammatical. L'analyse de tels énoncés s'arrête tôt, faute de prédiction cohérente, sans qu'il soit possible d'extraire les constituants partiels. En particulier si cette agrammaticalité se situe au début d'un énoncé, il sera impossible d'obtenir le moindre résultat partiel sur le reste de l'énoncé (même si celui-ci est par la suite grammatical).

Un effort important a consisté ces derniers années à développer un algorithme descendant pour les LTAG de complexité en $\mathcal{O}(n^6)$ vérifiant la propriété du préfixe valide [Nederhof97]. Un analyseur vérifiant cette propriété garantit que, lisant la chaîne à analyser de gauche à droite, la sous-chaîne actuellement reconnue constitue un préfixe valide du langage défini par la grammaire. L'intérêt est de pouvoir, en cas d'agrammaticalité, retrouver l'endroit sur la chaîne où se situe l'erreur. *A priori* cette propriété peut sembler utile pour les réparations d'analyse. Cependant, aucune application informatique n'exploite la propriété du préfixe valide. En effet, pour identifier quelle réparation d'analyse mettre en œuvre, il faut une frontière gauche du constituant après le point d'agrammaticalité, en plus d'une frontière droite du constituant au point d'agrammaticalité. Pour s'en convaincre, considérons simplement l'énoncé (37) qui présente une agrammaticalité due à un phénomène de reprise.

(37) *Jean prend le train de à Nancy.*

L'analyseur vérifiant la propriété du préfixe valide arrêtera son analyse après la lecture du cinquième mot, identifiant ainsi la position d'agrammaticalité. Cependant pour être capable de corriger l'analyse, c'est-à-dire effacer la préposition *de* et reprendre l'analyse à la seconde préposition, il faut savoir que le constituant après le point d'agrammaticalité est une préposition. Un tel analyseur n'est pas capable de fournir cette seconde frontière. La propriété du préfixe valide n'est donc pas suffisante pour assurer à ce type d'algorithme la robustesse nécessaire à l'analyse de la langue orale spontanée.

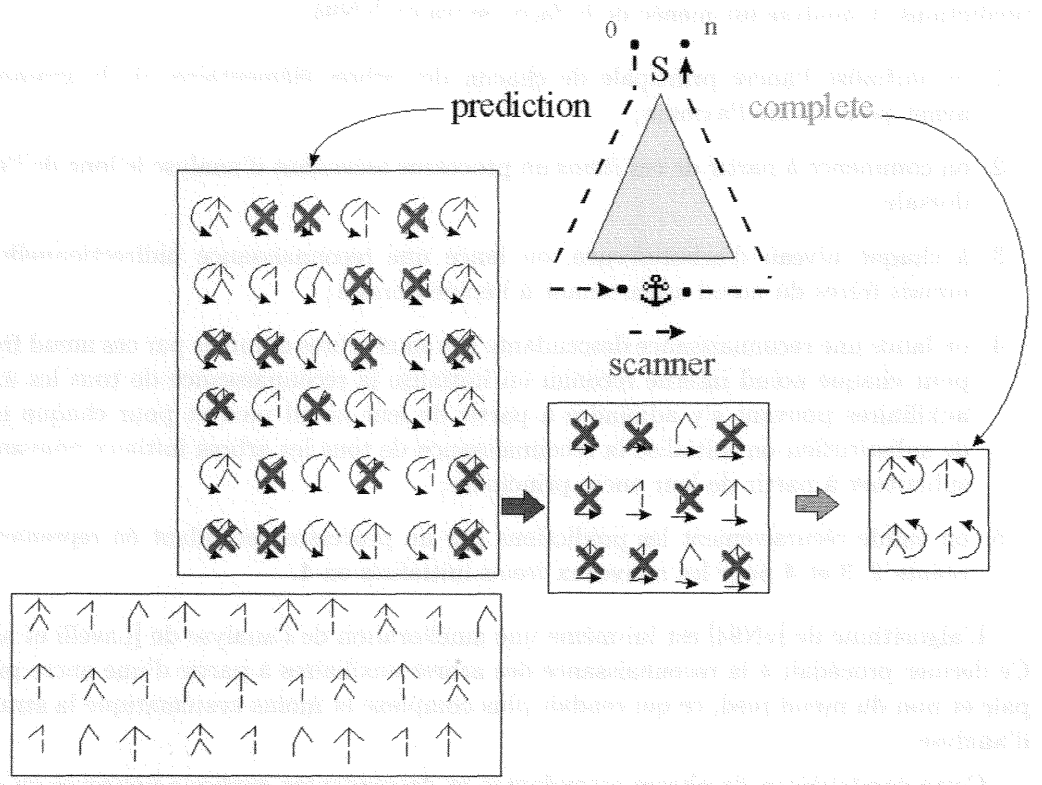


FIG. 3.4 – *Principe de l'algorithme de Earley : la phase de prédiction filtre l'espace de recherche, la phase de scanner valide certaines prédictions en fonction de la position des ancres, enfin complete valide la fin de la reconnaissance.*

3.4 Approches mixtes

3.4.1 Analyse guidée par l'épine dorsale

Certains travaux ont tenté d'apporter plus de souplesse et d'efficacité aux analyses ascendantes de LTAG [Lavelli et al.91] [vN94]. Tout d'abord, aucun des algorithmes de type CKY n'est bidirectionnel. L'adaptation de l'algorithme CKY suppose une forme normale de Chomsky, les sous-arbres étant successivement combinés deux à deux. Lorsque le parcours ascendant est impossible, la reconnaissance de l'arbre élémentaire s'arrête. Dans la version généralisée, les sous-arbres dominés par les fils successifs d'un même niveau d'arborescence sont reconnus de gauche à droite uniformément. Une première amélioration consiste à reconnaître ces fils de façon bidirectionnelle. Dans ces améliorations d'algorithmes ascendants, le nœud de départ n'est pas choisi arbitrairement : il s'agit du nœud appartenant à l'épine dorsale. L'épine dorsale désigne le chemin entre le nœud racine et le nœud pied pour les arbres auxiliaires, entre le nœud racine et l'ancre principale pour les arbres initiaux. Si on considère les choix linguistiques habituels de conception d'une grammaire LTAG, l'épine dorsale est donc le chemin qui relie la tête lexicale de l'arbre élémentaire au nœud racine. Il s'agit donc de débuter la reconnaissance de chaque niveau d'arborescence par la tête syntagmatique.

Après ce choix, les algorithmes de [Lavelli et al.91] et de [vN94] associent une étape de

prédictions. L'analyse est menée de la façon suivante [vN94] :

1. on initialise l'ancre principale de chacun des arbres élémentaires de la grammaire ayant pour racine l'axiome ;
2. on commence à partir de ces *items* un processus ascendant d'analyse le long de l'épine dorsale ;
3. à chaque niveau d'arborescence, on lance une reconnaissance bidirectionnelle des nœuds frères du nœud appartenant à l'épine dorsale ;
4. on lance une reconnaissance descendante des sous-arbres dominés par ces nœud frères : pour chaque nœud interne reconnu on initialise la reconnaissance de tous les arbres auxiliaires pouvant s'y adjoindre à partir de leur nœud pied et pour chaque nœud de substitution on initialise la reconnaissance de tous les arbres initiaux pouvant s'y substituer à partir de leur ancre principale ;
5. on valide récursivement les prédictions par un processus ascendant en reprenant les points 2, 3 et 4 pour les nouveaux *items* initialisés en 4.

L'algorithme de [vN94] est lui-même une amélioration de l'analyse de [Lavelli et al.91]. Ce dernier procédait à la reconnaissance des arbres auxiliaires à partir d'une ancre principale et non du nœud pied, ce qui rendait plus complexe et moins systématique la stratégie d'analyse.

Cette combinaison de phases ascendantes et descendantes explique pourquoi on parle habituellement, pour ces algorithmes, d'approche mixte. Le rôle des prédictions descendantes est de diminuer dynamiquement l'espace de recherche en filtrant uniquement les *items* pouvant intervenir dans une structure en cours de reconnaissance. Le rôle des phases de reconnaissance ascendantes est, d'une part, d'initialiser l'analyse de façon à maintenir une expansion d'îlots, résultat partiel d'analyse intéressant, et, d'autre part, de valider les *items* prédits.

Cette stratégie a une justification linguistique qui est celle globalement des algorithmes dirigés par les têtes. L'hypothèse est la suivante : c'est la tête prédicative syntaxique (ou sémantique suivant l'approche) qui détermine les arguments possibles, par conséquent des prédictions sur la base de la reconnaissance des têtes réduiront de façon plus importante l'espace de recherche que des prédictions sur la base d'autres catégories syntagmatiques. Les performances, rapportées dans [vN97], d'un analyseur de ce type avec d'autres algorithmes classiques, pour une grammaire hors contexte, semblent donner raison à cette idée sur le plan de l'efficacité.

Cette approche est donc intéressante, mais elle présente un manque de robustesse par exemple en cas d'absence de la tête syntagmatique (ellipses de tête), ce qui est fréquent en français, même à l'écrit. De plus un emploi robuste de l'analyse supposerait de considérer la plupart des catégories comme axiomes, ce qui limiterait l'efficacité des prédictions.

Aucun de ces algorithmes ascendants fonctionnant par extension d'îlots ne peut donner les segments grammaticaux les plus étendus, ce qui reste notre objectif. Ils sont adaptés à des analyses complètes d'énoncés. Enfin, en restant dépendant de la reconnaissance complète d'un niveau d'arborescence avant de pouvoir passer à un autre niveau, aucun ne tient compte de la topologie des arbres pendant l'analyse pour tenter d'étendre les analyses à leur maximum.

3.4.2 Algorithmes de Fabrice Issac

Fabrice Issac dans sa thèse [Issac98] propose un algorithme atypique d'analyse de LTAG ascendant qui peut être vu comme une tentative de prise en compte du domaine de localité étendu lors de l'analyse et comme une tentative de ne pas utiliser de technique tabulaire classique. Cependant l'approche proposée pose de sérieux problèmes pour plusieurs raisons :

- l'algorithme d'analyse proposé est exponentiel en espace et en temps, rendant toute application pratique sur une grammaire de taille moyenne et avec des phrases de plus d'une dizaine de mots impossible. En effet l'analyseur manipule des *items* relatifs à des arbres dérivés, sans positions de type arbre pointé. Les possiblités de combinaisons d'*items* sont contraints par des informations sur les arbres dérivés, et non simplement sur les arbres élémentaires (cas des algorithmes classiques). Les possiblités de combinaisons d'*items* dépendent donc de la dérivation en cours dont le nombre est au pire des cas exponentiel. La possibilité d'analyse polynomiale n'est ici pas satisfaite. L'auteur propose également un reconnaisseur de complexité polynomiale mais n'indique pas comment en faire un analyseur ;
- la prise en compte du domaine de localité étendu passe par un recalcul complet des positions des ancres pour chaque arbre dérivé lors du test d'une opération. Comme ces positions dépendent de la dérivation en cours, et qu'il n'y a pas de tabulation des sous-dérivations, on comprendra que l'algorithme mis en œuvre n'est pas trivial impliquant le parcours de plusieurs arbres élémentaires suivant un historique complexe. Pour des algorithmes classiques d'analyse de LTAG reposant sur des techniques tabulaires, une simple égalité des indices et de la catégorie courante permet de décider de l'inférence ou non d'un nouvel *item* ;
- pour le reconnaisseur, les *items* d'analyse sont stockés dans les nœuds des arbres élémentaires, rendant l'étape de validation d'une opération complexe par rapport à un CKY par exemple. De plus la gestion d'un agenda avec cette solution, la phase d'unification et l'extraction des éventuelles dérivations pour un éventuel analyseur déduit du reconnaisseur ne sont pas évoquées ;
- rien n'est proposé quant au blocage des dérivations superflues amenées par les analyses bidirectionnelles [Lavelli et al.91] ;
- aucune évaluation ni comparaison avec des algorithmes existants n'est proposée.

Le principal intérêt de cet algorithme est de fournir une analyse pour toutes les séquences contiguës grammaticales, permise par le recalcul systématique des positions de chaque hypothèse. Certains problèmes restent alors sans réponse. Comment fournir, parmi la masse de résultats partiels, des analyses à la fois cohérentes en termes de présence de co-ancres et seulement celles qui correspondent à une extension maximale? D'autre part comment aboutir au même type de résultat à l'aide d'un algorithme polynomial effectivement utilisable, sans recalcul de position pour chaque test d'une opération et fournissant les dérivations sous une complexité spaciale également polynomiale (classiquement une forêt partagée de dérivation)? Sur ce dernier point, il semble inévitable de passer par un algorithme tabulaire.

3.5 Limites des algorithmes classiques

Nous avons constaté que l'une des caractéristiques les plus importantes d'un analyseur syntaxique destiné à du langage naturel est sa capacité à fournir des résultats partiels

d'analyse les plus riches possibles lorsque l'analyse échoue. Les diagnostics d'erreurs, les réparations d'analyse et l'interprétation seront d'autant plus faciles si on peut exploiter les informations données par des arbres partiels d'analyse correspondants aux dérivations les plus étendues possibles. En utilisant un *chart* pour l'analyse syntaxique, les résultats partiels sont donnés par les *items* qui représentent un segment bien reconnu de l'énoncé. Les caractéristiques et les propriétés des résultats partiels sont données par l'invariant de l'algorithme d'analyse mis en œuvre. Nous allons voir dans cette partie en quoi l'invariant employé par les algorithmes d'analyse de LTAG traditionnels est limité pour nos objectifs de robustesse et d'identification des segments grammaticaux les plus étendus.

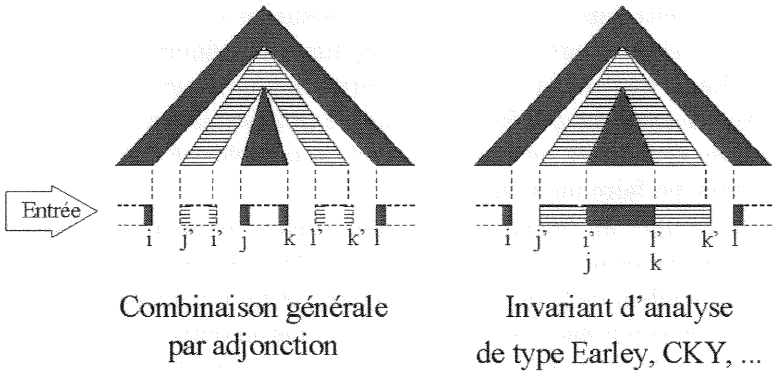


FIG. 3.5 – *Combinaison par adjonction : cas général sans point de connexité sur les chaînes dominées et cas imposé par l'invariant classique des algorithmes pour les LTAG.*

Pour les algorithmes de LTAG issus des algorithmes d'analyse de grammaires hors contexte, les invariants sont basés sur la reconnaissance de sous-arbres : chaque *item* représente une position sur un arbre élémentaire et un segment de la chaîne à analyser. L'invariant impose que le sous-arbre dominé par le nœud pointé associé à un *item* soit complètement analysé comme l'illustre la figure 3.5. Les règles employées combinent des sous-arbres complètement analysés en de nouveaux sous-arbres préservant toujours cet invariant.

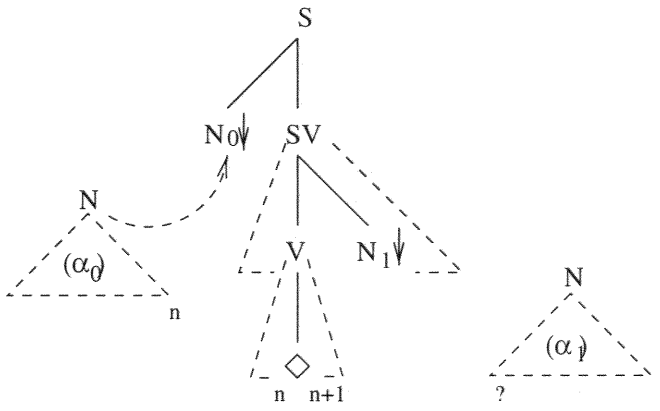


FIG. 3.6 – *Parcours ascendant d'un arbre et invariant lié au sous-arbre analysé.*

Sur la figure 3.6, nous avons indiqué différents sous-arbres ayant été complètement analysés afin d'analyser l'ensemble de l'arbre élémentaire représenté. Supposons maintenant que l'énoncé à analyser soit agrammatical parce qu'un constituant non prévu soit présent sur la droite (à la position notée par ?). Le sous-arbre α_1 sur la droite n'est pas adjacent à l'ancre de l'arbre élémentaire et donc un rattachement sur le nœud de substitution N_1 est impossible. Le sous-arbre dominé par le nœud SV ne peut pas être complètement reconnu et donc le processus d'analyse de l'ensemble de l'arbre élémentaire doit s'arrêter, même si le sous-arbre α_0 peut encore se combiner par adjacence au nœud de substitution N_0. En considérant une analyse bidirectionnelle de type CKY ou de type guidée par les Têtes [vN94], lorsqu'un constituant non prévu par la grammaire apparaît, l'invariant classique n'est plus respecté et l'analyse est stoppée des deux côtés du sous-arbre en cours d'analyse. Cet arrêt a lieu même lorsqu'il est possible de continuer l'analyse d'un côté. La bidirectionnalité n'étant pas totalement exploitée, le résultat correspond à des analyses partielles plus limitées que ce qu'il serait possible de réaliser.

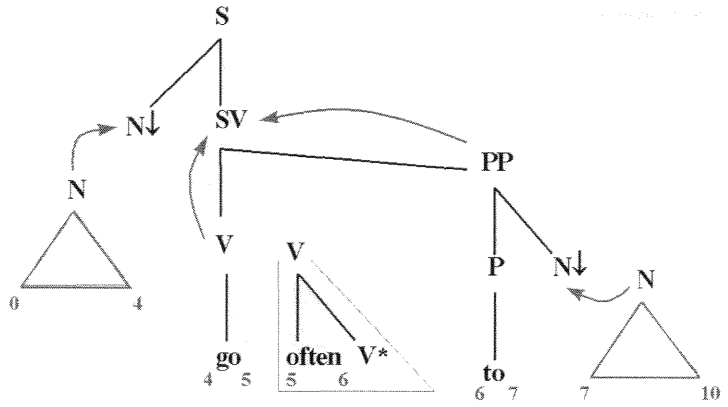


FIG. 3.7 – *Un autre exemple de parcours ascendant d'un arbre et d'invariant lié au sous-arbre analysé.*

La figure 3.7 donne un autre exemple des limites associées à cet invariant. Supposons que dans une grammaire LTAG de l'anglais, on ait interdit aux adverbes d'apparaître après le verbe. Supposons que l'on ait à analyser un énoncé où, pour une raison de distorsion de l'oral par exemple, un adverbe apparaît après le verbe comme *often* sur cet exemple. L'arbre élémentaire associé à cet adverbe ne pourra donc pas se rattacher à l'arbre associé au prédicat verbal *go to*. Cette incomplétude rendra impossible la remontée au nœud *SV* de cet arbre, puisque le sous-arbre dominé par ce nœud n'a pas été entièrement reconnu. L'analyse de l'arbre élémentaire associé au prédicat verbal prend donc fin. Si une technique robuste d'extension des co-ancres est mise en œuvre, il sera possible de rattacher le groupe nominal objet à l'arbre élémentaire prédicatif. Par contre, le rattachement du groupe nominal sujet ne sera jamais évalué, bien qu'il soit tout-à-fait cohérent par rapport aux indices considérés. Le segment s'étendant de la position d'indice *0* à la position d'indice *5* est tout-à-fait grammatical compte tenu de la grammaire LTAG considérée, mais ne sera donc pas reconnu. Pourtant un tel rattachement du groupe nominal est intéressant à plus d'un titre, tout particulièrement :

– l'identification d'un sujet ou d'un objet supplémentaire au cours d'une analyse permet

de fournir des résultats plus riches pour l'interprétation de l'énoncé ;
- dans le cas d'une analyse stochastique, un rattachement supplémentaire permet de contraindre de façon plus importante les probabilités de rattachements ultérieurs.

Nous constatons donc qu'en se basant sur un invariant d'analyse classique, nous ne pouvons pas fournir les segments grammaticaux les plus étendus. La principale raison est que l'expansion de l'analyse n'est pas réellement bidirectionnelle. Lorsqu'une analyse s'arrête, elle le fait parce qu'une agrammaticalité apparaît d'un coté d'expansion de l'analyse, même s'il est encore possible de progresser de l'autre côté. Au mieux, l'analyse peut s'étendre de l'autre côté jusqu'à la reconnaissance de tous les frères, c'est le cas de l'algorithme de type *head corner*, mais ne pourra poursuivre au niveau père. Leur objectif étant avant tout de diminuer la complexité d'analyse, ces algorithmes se focalisent sur les analyses complètes qui doivent être obtenues avec une complexité la moins importante possible, sans donner une grande importance aux résultats partiels. La partie suivante propose un nouvel algorithme qui, à l'inverse, tente de fournir les résultats partiels les plus étendus tout en assurant leur cohérence.

Troisième partie

Techniques d'analyse guidée par la connexité de Grammaires Lexicalisées d'Arbres

Chapitre 1

Un nouvel algorithme ascendant : l'analyse par connexité

L'expression de contraintes à l'aide d'arbres élémentaires lexicalisés est plus riche qu'avec un ensemble de règles de réécriture hors contexte. Un arbre élémentaire associe en fait à une ancre lexicale l'équivalent d'une suite de réécriture hors contexte. Cette constatation est le résultat des deux propriétés fondamentales des grammaires LTAG dont il nous semble important de tenir compte durant l'analyse :

- le domaine de localité étendu : des contraintes structurelles peuvent porter sur l'ensemble de l'arbre élémentaire analysé et au-delà d'une simple règle de réécriture hors contexte. Il nous semble important de pouvoir exploiter l'ensemble du domaine de localité durant l'analyse, en particulier les co-ancres ;
- la lexicalisation : il est possible de se limiter à l'analyse de la sous-grammaire contenant uniquement les arbres élémentaires ancrant un des mots de l'énoncé.

Nous avons également insisté sur l'importance d'extraire d'un énoncé un maximum d'information, comme par exemple l'ensemble des constituants compte tenu d'une analyse locale, ceci même si l'analyse complète de l'énoncé échoue. Cette approche se révèle contradictoire avec l'utilisation de prédictions durant l'analyse. Les prédictions accélèrent habituellement un analyseur. Cependant, comme expliqué par exemple dans [Magerman et al.92], si l'utilisation de prédictions de type Earley améliore les performances moyennes d'analyse en temps, elle présente un sérieux impact sur la robustesse des analyses d'énoncés incomplets et agrammaticaux. Or ce type d'énoncé est très commun en langue naturelle, et c'est sur cette constatation que [Magerman et al.92] rejète ce type d'algorithme pour les systèmes de dialogue.

Nous avons présenté, dans la section 3.5 de la partie précédente, la principale limite de l'invariant mis en œuvre dans les algorithmes classiques d'analyse de LTAG du point de vue de la robustesse. Notre objectif est de centrer l'analyse sur les îlots bien reconnus plutôt que sur les sous-arbres bien reconnus, de façon à permettre une extension réelement bidirectionnelle des hypothèses d'analyse. Nous verrons alors que, dans la mesure où les techniques tabulaires imposent une condition d'adjacence sur les *items* à combiner, nous pouvons prendre en compte la topologie des arbres de manière à diminuer la complexité moyenne d'analyse. Ceci repose sur un nouveau niveau de granularité pour les linéarisations d'arbres élémentaires nommé *parcours connexes*.

Les techniques d'analyse que nous allons présenter ici peuvent s'appliquer à tout type de Grammaire d'Arbres Lexicalisées. L'algorithme spécifié dans ce chapitre permet l'ana-

lyse de LTAG et de TIG lexicalisées, prenant en compte la complexité de ces deux types de grammaires. Rappelons qu'une Grammaire Lexicalisée d'Arbres Insérés (TIG) est une grammaire lexicalisée d'arbres qui ne contient aucun arbre auxiliaire englobant et qui suppose que ce type d'arbres ne se construit pas dynamiquement au cours de l'analyse. Comme expliqué dans la section précédente, l'utilisation d'une TIG permet la réduction de la complexité d'analyse au pire des cas en (n^3) car les dérivations sont alors équivalentes à celles d'une grammaire hors contexte. La prise en compte de TIG supposera ainsi que la complexité d'analyse effectivement mise en œuvre soit en (n^3), ce qui n'est pas le cas de l'algorithme de type Earley proposé dans [Schabes94] pour les LTAG, qui présente une complexité en (n^4) lors de l'analyse de TIG.

1.1 Techniques d'analyse par connexité

1.1.1 Grammaire lexicalisée d'arbres : notations

Nous notons un nœud de substitution avec sa catégorie et la marque $\downarrow$, un nœud interne sans marque spécifique, un nœud racine avec la marque $^{\triangle}$ et les ancres par la marque lexicale $\Diamond$. Nous notons $*_g$ (resp. $*_d$) le nœud pied d'un arbre auxiliaire droit (resp. gauche), et $*$ le nœud pied d'un arbre auxiliaire englobant.

De façon à simplifier les explications, nous ne considérons pas les contraintes de non-adjonction sur les nœuds. Notons que leur prise en compte ne pose pas de problème particulier.

1.1.2 Représentation d'arbres élémentaires linéarisés par automates d'états finis

La linéarisation d'un arbre peut être représentée par un Automate à Etats Finis (FSA) comme illustré sur la figure 1.1. La linéarisation donne simplement une suite de catégories que nous associons à une série de transitions d'un automate linéaire. Les états de l'automate correspondent dans l'arbre élémentaire à un parcours d'un nœud vers un nœud voisin. Tout parcours d'arbre (de gauche à droite, bidirectionnel à partir d'une ancre, etc.) peut alors se réaliser sur cet automate en parcourant simplement ses états et ses transitions.

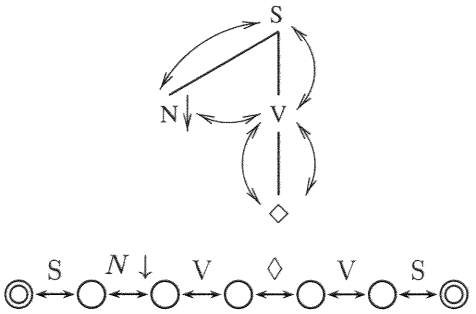


FIG. 1.1 – *FSA représentant l'arbre élémentaire correspondant à la forme normale du contexte verbal intransitif.*

Nous considérons les définitions et notations suivantes :

- transition de l'automate est annotée avec une catégorie syntaxique. Chaque catégorie correspondant à des nœuds non feuilles apparaît deux fois dans la liste des transitions.

Les deux transitions liées à cette catégorie encadrent alors les transitions correspondant aux nœuds dominés. De manière à simplifier les notations, les transitions sont notées par la catégorie annotée ;

- les transitions sont bidirectionnelles de façon à pouvoir réaliser un parcours bidirectionnel à partir de n'importe quel état ;
- en considérant un type de parcours, les transitions sont alors unidirectionnelles, et l'automate devient acyclique.

1.1.3 Equivalence entre état d'automate et arbre pointé

Étant donné le principe de création des automates, les arbres pointés, utilisés par exemple dans [Schabes94] ou [Shieber et al.95], sont équivalents aux états de ces automates ou encore aux règles de production pointées de [Nederhof97]. Cette équivalence est illustrée par la figure 1.2.

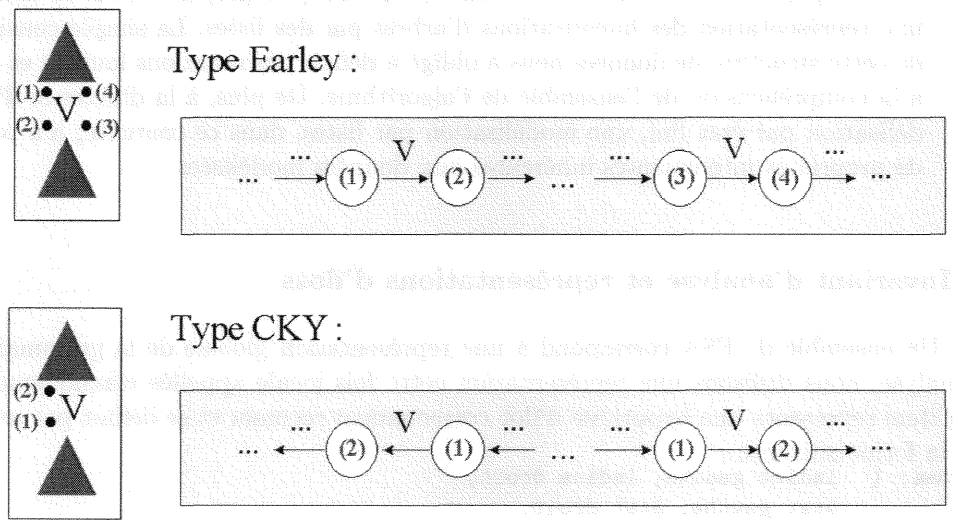


FIG. 1.2 – *Équivalence entre arbre pointé et état d'automate.*

En conséquence, n'importe quel algorithme fondé sur la notion d'arbre ou de règle pointé (c'est-à-dire la quasi-totalité des algorithmes proposés pour les LTAG) peut s'appuyer sur de tels automates, qui sont des représentations équivalentes à des arbres élémentaires. Si il y a équivalence, on peut se demander pourquoi considérer une représentation par automates plutôt que la notion, maintenant classique pour l'analyse de LTAG, d'arbres ou de productions pointés? La réponse repose en fait sur trois points :

1. Considérer l'informatique comme une science est un débat susceptible de faire couler beaucoup d'encre. Cependant quelles qu'en soient les conclusions, parvenir à des modélisations de structures de données permettant une implantation informatique effective, n'en suppose pas moins une étude rigoureuse des principes élémentaires mis en œuvre. L'algorithmique, puisque c'est de cela qu'il s'agit, peut se définir comme la description de données et de processus complexes (traitement de l'image, du son, de texte, etc.) en un ensemble de données simples sur lesquels on peut appliquer une série d'actions élémentaires. Les techniques à états finis sont parmi les représentations

les plus connues et efficaces, tandis que la notion d'arbres pointés est plus spécialisée et d'un niveau complexe.

2. Ce second point est en fait la conséquence logique du précédent : identifier les principes et les données élémentaires mis en jeu dans un processus complexe n'est pas seulement une question d'optimisation des traitements ou d'élégance dans une démarche de modélisation. Une telle identification permet, par la suite, d'isoler de façon simple des propriétés difficilement perceptibles à un niveau de description complexe. Ainsi, nous verrons que la représentation d'une grammaire lexicalisée d'arbres en automates permet de réaliser une compaction efficace de la grammaire par simple minimalisation des automates, compaction difficile à entreprendre sur des arbres, et permet de considérer plus facilement la composante d'une grammaire LTAG qu'il est possible de capter par états finis.
3. Enfin, cette représentation permet une formalisation plus simple et plus lisible de l'algorithme que nous présentons ici : deux versions préliminaires de cet algorithme ont été présentées successivement dans [Lopez98b] et [Lopez98c] en se fondant sur une représentation des linéarisations d'arbres par des listes. La simple considération de cette structure de données nous a obligé à définir des notations lourdes et nuisibles à la compréhension de l'ensemble de l'algorithme. De plus, à la différence d'une modélisation par état fini, une modélisation par listes, dans ce contexte, n'a pas révélé de propriétés intéressantes inhérentes aux données modélisées.

1.1.4 Invariant d'analyse et représentations d'îlots

Un ensemble de FSA correspond à une représentation globale de la grammaire. Pour l'analyse, nous utilisons une représentation cette fois locale appellée classiquement *item*. Un *item* représente une hypothèse d'îlot correctement reconnu et se définit par un 7-uplet de la forme suivante :

```
item : (  indice gauche, indice droit,
          état gauche, état droit,
          indice gauche du nœud pied,
          indice droit du nœud pied, état étoile )
```

Les deux premiers indices sont les limites par rapport à la chaîne à analyser de l'îlot représenté par l'*item* (voir figure 1.3). Au cours de la phase d'initialisation, on construit un item pour chaque ancre présente dans la chaîne d'entrée. Un *item* contient également deux pointeurs sur des états d'un même automate correspondant à l'extension courante de l'îlot sur la gauche et sur la droite. On représente également dans un *item*, seulement si nécessaire, deux indices additionnels pour la position du nœud pied d'un arbre auxiliaire englobant et un pointeur sur l'état *étoile*, similaire à l'algorithme de Schabes [Schabes94], correspondant au nœud où l'adjonction englobante courante a été prédite. Lorsque l'*item* ne correspond pas à un arbre auxiliaire englobant, ces champs du 7-uplets restent indéfinis, car inutiles, et ne sont pas représentés pour alléger les notations.

Cette représentation maintient l'invariant suivant illustré figure 1.3 : un *item* de la forme (p,q,σ_L,σ_R) spécifie le fait qu'un arbre linéarisé représenté par une FSA Δ est complètement analysé entre les états σ_L et σ_R de Δ et couvre la chaîne d'entrée entre les indices p et q. Plus aucun rattachement ne peut avoir lieu sur les nœuds situés entre les ancres du segment de p à q-1.

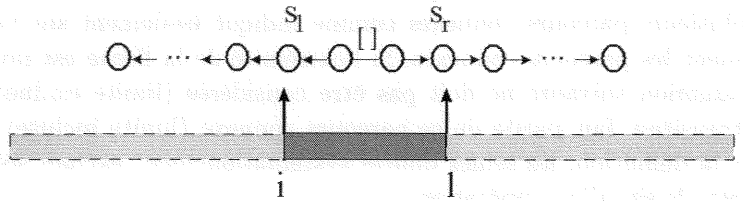


FIG. 1.3 – *Correspondance entre item d'analyse et îlot.*

1.1.5 Parcours connexes

En considérant qu'un automate représente la linéarisation d'un arbre élémentaire, nous pouvons définir un *parcours connexe* comme une partie de cet automate correspondant à la liste des nœuds parcourus successivement jusqu'à atteindre un nœud de substitution, un nœud pied ou un nœud racine (transition alors incluse au parcours connexe) ou une ancre (transition exclue). Un parcours connexe est un niveau de représentation intermédiaire entre le nœud et l'arbre complet pour un arbre linéarisé : tout arbre élémentaire (ou dérivé) peut être représenté comme une liste de parcours connexes. Nous allons voir que la notion de parcours connexe permet durant l'analyse de prendre en compte la topologie des arbres élémentaires et de situer les nœuds significatifs impliqués dans le rattachement de deux arbres.

Pour expliquer plus intuitivement la définition d'un parcours connexe et illustrer leur intérêt, considérons l'exemple des deux arbres élémentaires de la figure 1.4 et les parcours gauche-droite en *profondeur d'abord* associés.

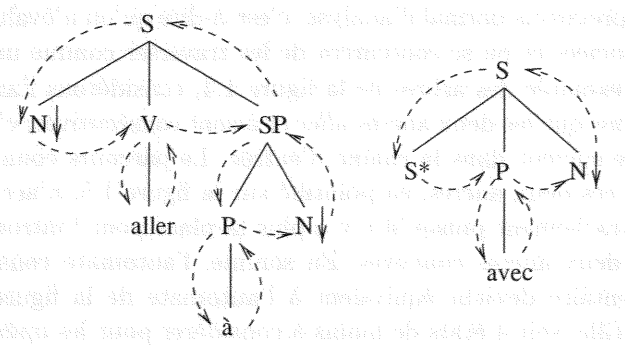


FIG. 1.4 – *Exemple d'un parcours de gauche à droite d'un arbre élémentaire.*

Chaque nœud interne ou racine apparaît deux fois parmi la liste des nœuds traversés, encadrant les nœuds qu'il domine. Par exemple, à partir des parcours de la figure 1.4, nous obtenons la liste suivante de nœuds :

$$1 : S^{\triangle},\ N{\downarrow},\ V,\ \texttt{aller},\ V,\ SP,\ P,\ \texttt{à},\ P,\ N{\downarrow},\ SP,\ S^{\triangle}$$

$$2 : S^{\triangle},\ S{*_d},\ P,\ \texttt{avec},\ P,\ N{\downarrow},\ S^{\triangle}$$

Pour représenter ces deux arbres élémentaires, nous introduisons donc, déduits de ces deux listes, les deux automates de la figure 1.5.

En appliquant la définition des parcours connexes, nous pouvons segmenter un automate en plusieurs parcours connexes comme indiqué également sur l'exemple 1.5. Les flèches indiquent les parcours connexes. Si l'extrémité de la flèche est inversée, cela signifie que la transition suivante ne doit pas être considérée (limite exclue), si la flèche est normale la transition fait partie de ce parcours connexe (limite incluse). En fait, comme indiqué dans la définition, les seules limites systématiquement exclues sont les ancres, qui ne peuvent être le site d'une opération.

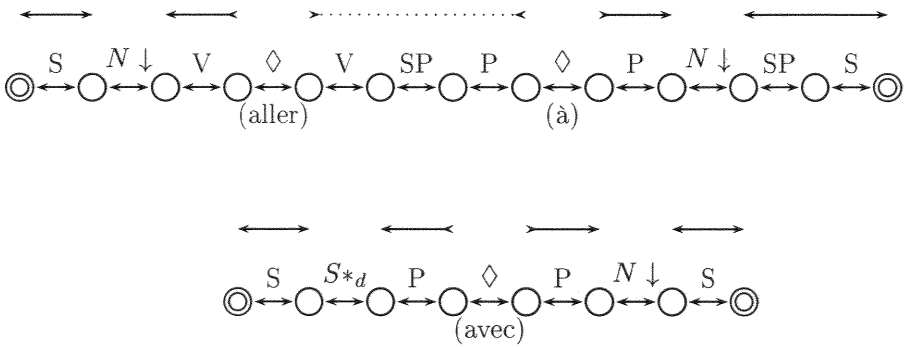


FIG. 1.5 – *Parcours connexes associés aux FSA représentant les linéarisations des arbres élémentaires de la figure 1.4.*

Le principal avantage d'introduire la notion de parcours connexe au cours de l'analyse est le suivant : les parcours connexes situés entre deux ancres consécutives deviennent inutiles à considérer durant l'analyse, parce qu'il n'y a tout simplement plus de place pour une opération sur un des états qu'ils contiennent. De tels parcours connexes peuvent être « éliminés » du processus normal d'analyse, c'est-à-dire qu'on n'évaluera pas les possibilités d'opérations associées et on se contentera de les traverser comme un automate ordinaire.

Reprenons l'exemple des arbres de la figure 1.4, considérons l'arbre élémentaire de *aller_à* et supposons que les deux ancres *aller* et *à* sont consécutives, c'est-à-dire que les mots correspondant se suivent dans la chaîne d'entrée. Le parcours connexe de l'arbre élémentaire situé entre ces deux ancres, en pointillé sur la figure 1.5, n'accueillera jamais aucune opération de rattachement puisqu'il n'y a plus la place pour l'introduction d'une nouvelle ancre entre ces deux ancres connexes. En somme, l'automate considéré durant l'analyse de l'arbre élémentaire devient équivalent à l'automate de la figure 1.5 sans le parcours connexe en pointillé, soit 4 états de moins à considérer pour les opérations d'adjonction et de substitution.

Une telle élimination sera réalisée à chaque fois que deux ancres sont connexes, c'est-à-dire à chaque fois qu'un rattachement sera réalisé en se basant sur des techniques tabulaires classiques et sur les linéarisations d'arbres définies. Comme nous exploitons cette propriété, nous disons que notre algorithme est guidé par la connexion des ancres. Cette opération d'élimination porte le nom de *réduction de co-ancres* et permet de considérer moins d'états (donc moins de nœuds) durant l'analyse et donc moins d'hypothèses à tester.

Nous utilisons les notations suivantes :

- $head(\Gamma)$ (resp. $tail(\Gamma)$) donne la première transition partant de (resp. arrivant à) l'état le plus à gauche (resp. le plus à droite) du parcours connexe Γ ;
- le parcours connexe passant par l'état σ_D est noté Γ_σ. Γ_D est le parcours connexe

partant de σ_D et s'étendant jusqu'à $tail(\Gamma_\sigma)$ si σ_D est point d'expansion droite ($D = R$), jusqu'à $head(\Gamma_\sigma)$ si σ_D est point d'expansion gauche ($D = L$) ;
- $next(\Gamma)$ (resp. $previous(\Gamma)$) donne le premier état d'un parcours connexe après (resp. avant) Γ suivant un parcours gauche-droite de l'automate ;
- $next(N)$ (resp. $previous(N)$) donne l'état après (resp. avant) la transition N.

1.2 Système de règles d'inférence

Le processus de dérivation peut être vu comme l'application successive d'une série de règles d'inférence qui utilisent et produisent des *items*. Nous employons des règles d'inférence similaires à celles de [Schabes94] ou [Sikkel97] pour spécifier notre algorithme. La règle d'inférence ci-dessous a la signification suivante : si $item_1$ et $item_2$ sont présents dans le *chart* et si les *conditions* sont satisfaites alors ajouter $item_3$ dans le chart *si nécessaire*.

$$\frac{item_1 \qquad item_2}{\text{add} \quad item_3} \quad (\text{ conditions })$$

Nous allons présenter les différentes règles de l'algorithme ascendant proposé. Le principe général est le suivant : un des deux *items* arguments peut-être considéré comme le foncteur de l'opération, on considère alors successivement les nœuds de ses parcours connexes droit ou gauche. Il n'y a alors plus qu'à tester si un autre *item* adjacent satisfait les conditions pour être opérande d'une des règles. Si on considère pour chaque *item* foncteur uniquement le parcours connexe droit, on obtiendra un algorithme ascendant procédant de gauche à droite. Il serait possible d'être prédictif selon une stratégie de type *coin gauche* une fois le premier *item* considéré. Réciproquement si on ne considère pour l'*item* foncteur que le parcours connexe gauche, on obtient un algorithme ascendant fonctionnant par extension des îlots de droite à gauche. En considérant à la fois parcours connexe droit et gauche pour les *items* foncteurs, ce qui suppose en pratique d'implanter chaque règle et son symétrique, on obtient une analyse ascendante bidirectionnelle. C'est ce dernier emploi que nous avons retenu en pratique.

Chaque règle a pour rôle d'étendre les positions sur l'arbre linéarisé plutôt que de monter sur l'*épine dorsale* des arbres élémentaires comme [Lavelli et al.91] ou [vN94] et de considérer des sous-arbres d'un même niveau d'une règle de production. Dans l'algorithme que nous proposons il est possible de poursuivre l'analyse à un nœud alors que l'ensemble de ses fils ne sont pas encore entièrement évalués. Ceci est possible parce que nous analysons des arbres élémentaires et que nous connaissons, grâce à l'encodage d'un tel arbre, le père de chaque nœud interne. Même si une analyse est interrompue d'un coté de l'*épine dorsale*, notre algorithme permet d'atteindre le nœud racine par l'autre coté de l'*épine dorsale* (à supposer bien sûr qu'il n'y ait pas d'agrammaticalité se présentant de ce second côté). Un algorithme classique stopperait complètement l'analyse pour cet arbre élémentaire.

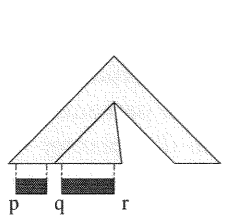
1.2.1 Initialisation

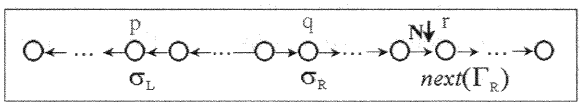
Le processus d'initialisation est classique aux algorithmes ascendants présentés dans le chapitre précédent. Étant donnée une chaîne d'entrée, on sélectionne, dans un premier temps, la sous-grammaire pouvant ancrer les mots effectivement présents. Il faut vérifier que les co-ancres sont elles aussi présentes dans l'énoncé et que l'ordre des ancres dans chaque arbre élémentaire est cohérent compte tenu de l'ordre des mots de l'énoncé. On crée alors un *item* par ancre et co-ancre possible, que l'on range dans un *chart*.

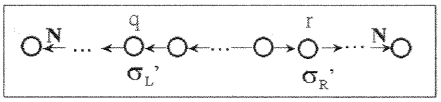
1.2.2 Substitution

La substitution est une dérivation hors contexte et ne présente pas de difficulté particulière pour l'analyse. Deux règles permettent de représenter un rattachement par substitution par extention respectivement vers la droite (cas 1) et vers la gauche (cas 2) sur la table 1.1. Les configurations sont illustrées à l'aide d'arbres élémentaires stylisés et de segments de chaînes reconnus. Dans le premier cas, le parcours connexe droit d'un premier îlot s'étendant entre p et q se termine par une transition correspondant à un nœud de sustitution $N\downarrow$. Un second îlot s'étendant de q à r et correspondant à un arbre initial α correctement reconnu est substituable en $N\downarrow$. Un arbre initial est correctement reconnu lorsque les parcours connexes gauche et droit aboutissent respectivement à l'état initial et final de l'automate Δ représentant α. On peut alors considérer un rattachement par substitution résultant en un îlot s'étendant de p à r, avec comme position d'extention gauche celle du premier *item* et comme position d'extension droite le premier état suivant la transition $N\downarrow$ dans la linéarisation. Cet état est aussi le premier état du parcours connexe suivant le parcours connexe droit du second *item*, c'est-à-dire $next(\Gamma_R)$. Les positions relatives aux automates des deux arbres élémentaires impliqués sont également illustrées.

Règles pour la substitution

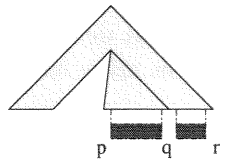


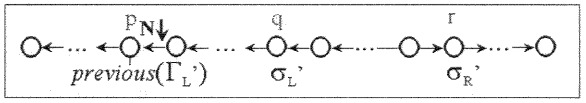


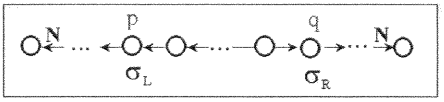


Substitution cas 1 :

$$\frac{(p,q,\sigma_L,\sigma_R) \qquad (q,r,\sigma_L',\sigma_R')}{(p,r,\sigma_L,next(\Gamma_R))} \quad \left(\begin{array}{ll} tail(\Gamma_R) = N\downarrow & (\sigma_L' \wedge \sigma_R') \in \Delta \qquad \Delta \in I \\ head(\Gamma_L') = N^{\Delta} & tail(\Gamma_R') = N^{\Delta} \end{array} \right)$$







Substitution cas 2 :

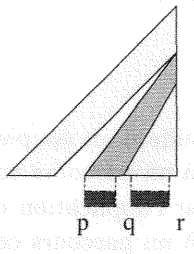
$$\frac{(p,q,\sigma_L,\sigma_R) \qquad (q,r,\sigma_L',\sigma_R')}{(p,r,previous(\Gamma_L'),\sigma_R')} \quad \left(\begin{array}{ll} head(\Gamma_L') = N\downarrow & (\sigma_L \wedge \sigma_R) \in \Delta \qquad \Delta \in I \\ head(\Gamma_L) = N^{\Delta} & tail(\Gamma_R) = N^{\Delta} \end{array} \right)$$

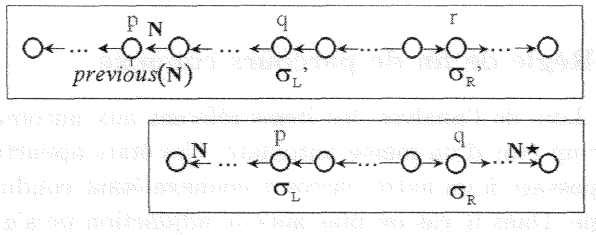
TAB. 1.1 – *Règles d'inférence pour la substitution.*

1.2.3 Adjonctions correspondant à des dérivations hors contextes

L'adjonction gauche et droite (ou furcation gauche et droite) sont également des dérivations hors contextes. Le principe est ici le même que celui mis en œuvre pour la substitution, voir table 1.2. L'analyse mise en œuvre pour ces opérations ne nécessite que de considérer des sous-chaînes adjacentes. L'attachement suivant ce type de dérivation donne lieu à la production d'un nouvel *item* représentant une extension d'îlot. Ce nouvel *item* réfère notamment à l'état suivant la transition qui correspond au nœud site d'adjonction, c'est-à-dire $previous(N)$ pour l'adjonction gauche et $next(N)$ pour l'adjonction droite, exprimant ainsi la progression de l'analyse.

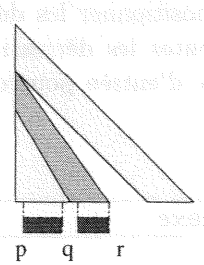
Règles pour les adjonctions hors contextes

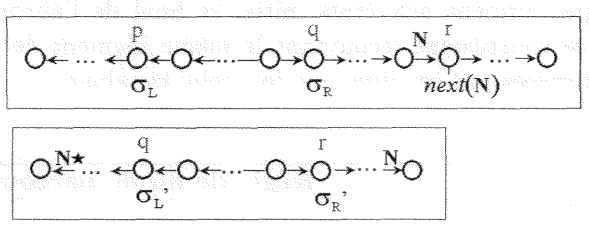




Adjonction gauche :

$$\frac{(p,q,\sigma_L,\sigma_R) \quad (q,r,\sigma_L',\sigma_R')}{(p,r,previous(N),\sigma_R')} \quad \begin{array}{l}(\ \exists(N^{\triangle} \vee N) \in \Gamma_L' \quad (\sigma_L \wedge \sigma_R) \in \Delta \quad \Delta \in A \\ head(\Gamma_L) = N^{\triangle} \quad tail(\Gamma_R) = N*_g\ )\end{array}$$





Adjonction droite :

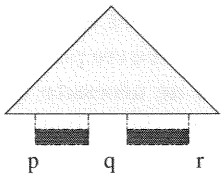
$$\frac{(p,q,\sigma_L,\sigma_R) \quad (q,r,\sigma_L',\sigma_R')}{(p,r,\sigma_L,next(N)} \quad \begin{array}{l}(\ \exists(N^{\triangle} \vee N) \in \Gamma_R \quad (\sigma_L' \wedge \sigma_R') \in \Delta \quad \Delta \in A \\ head(\Gamma_L') = N*_d \quad tail(\Gamma_R') = N^{\triangle}\ )\end{array}$$

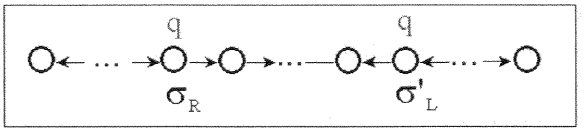
TAB. 1.2 – *Règles d'inférence pour les opérations d'adjonction gauche et droite.*

1.2.4 Réduction de co-ancres

La règle de *réduction des co-ancres*, table 1.3, permet de combiner deux îlots appartenant au même arbre. Si deux îlots appartiennent à un même sous-arbre, c'est qu'ils proviennent de l'extension de deux co-ancres d'un même arbre élémentaire. En pratique, cette règle s'applique lorsque deux *items* référant à un même automate présentent une extension droite pour le premier et une extension gauche pour le second égales.

Règle pour la réduction des co-ancres





Réduction des co-ancres :

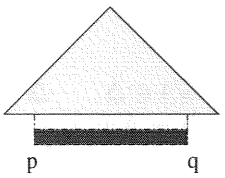
$$\frac{(p,q,\sigma_L,\sigma_R) \qquad (q,r,\sigma_L{}',\sigma_R{}')}{(p,r,\sigma_L,\sigma_R{}')} \quad (\ \Gamma_R = \Gamma_L{}' \ )$$

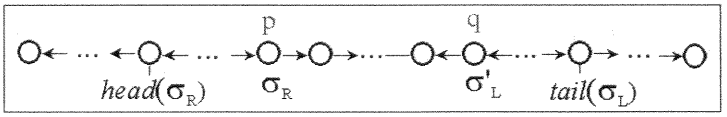
TAB. 1.3 – *Règle d'inférence pour la réduction des co-ancres.*

1.2.5 Règle de fin de parcours connexe

Lors de l'analyse, les *items* réfèrent aux automates à l'aide d'un couple de pointeurs sur un état d'un même automate. Ces états appartiennent à un parcours connexe donné, le passage à un autre parcours connexe étant conditionné par l'application d'une nouvelle règle. Dans le cas où plus aucune adjonction ne s'applique à un parcours connexe donné, l'état référé peut être un état quelconque de ce parcours connexe. Afin de permettre une factorisation efficace des *items*, nous introduisons la règle présentée table 1.4, proche d'une règle de *non-adjonction* pour les algorithmes classiques, qui permet de pousser les positions par rapport à l'automate jusqu'à la fin du parcours connexe courant. Par exemple, pour un constituant complètement reconnu, cette règle permet de positionner les deux pointeurs respectivement aux états initial et final de l'automate. Toutes les dérivations possibles de ce constituant recouvrant le même segment de la chaîne d'entrée pourront donc être factorisées par cet *item* une fois celui-ci tabulé.

Règle de fin de parcours connexe





Fin de parcours connexe :

$$\frac{(p,q,\sigma_L,\sigma_R)}{(p,r,head(\sigma_L),tail(\sigma_R))}$$

TAB. 1.4 – *Règle d'inférence pour la fin de parcours connexe.*

Nous avons considéré ici toutes les opérations nécessaires pour mener toutes les dérivations hors contextes. En particulier, ce système de règles est en mesure de mener des

analyses à partir d'une TIG avec une complexité au pire des cas en $\mathcal{O}(n^3)$, parce qu'aucun arbre auxiliaire englobant n'est construit dynamiquement durant l'analyse. En effet, on constate que les différentes règles présentent toutes un nombre maximum de positions libres sur la chaîne d'entrée de 3.

1.2.6 Adjonctions légèrement dépendantes du contexte

L'analyse générale de LTAG doit résoudre le problème des chaînes non contigus liées aux adjonctions englobantes. En fait une dérivation de ce type combine plusieurs sous-chaînes non adjacentes. Ceci implique qu'une étape de dérivation correspond à plusieurs étapes de reconnaissance sur la chaîne d'entrée, à la différence des dérivations hors-contextes où une dérivation peut correspondre à une seule étape d'analyse.

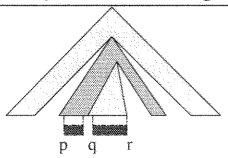
Un arbre auxiliaire englobant comprenant un nœud pied peut dominer une paire de chaînes non adjacentes, une sur la droite et une sur la gauche du nœud pied. L'analyse de ce type de rattachement se fait en deux étapes : tout d'abord l'initialisation des positions possibles du nœud pied, puis la reconnaissance complète du sous-arbre dominé par le nœud interne ou racine sur lequel porte l'adjonction. Cette seconde étape emploie deux indices supplémentaires pour certains *items* afin de représenter la position du nœud pied.

L'adjonction englobante est réalisée de différentes manières suivant la position des ancres de l'arbre auxiliaire et de l'arbre site de l'adjonction :

1. Le premier cas, table 1.5, s'applique si l'arbre auxiliaire comporte au moins deux ancres présentes respectivement à droite et à gauche de son épine dorsale et si une ancre au moins est présente dans le sous-arbre devant être dominé par le nœud pied. Ce cas passe par une initialisation d'abord partielle du nœud pied à l'aide de l'extension d'une des deux ancres de l'arbre auxiliaire et de la reconnaissance du sous-arbre dominé par le nœud pied. Une fois l'extension de la seconde ancre de l'arbre auxiliaire complétée, l'opération sera validée. Le pointeur vers la transition N_0 permet de retrouver la partie supérieure de l'arbre sur lequel porte l'adjonction.
2. Le second cas, table 1.6, suppose également que l'arbre auxiliaire comporte au moins deux ancres présentes respectivement à droite et à gauche de son épine dorsale, mais cette fois qu'aucune ancre n'est présente dans le sous-arbre dominé par le nœud pied. Dans ce cas, l'initialisation du nœud pied se fera en deux temps, grâce à l'expansion tout d'abord d'une première ancre de l'arbre auxiliaire jusqu'à adjacence avec un îlot de la partie supérieure de l'arbre sur lequel porte l'adjonction. Le sous-arbre dominé par le nœud pied, qui ne comporte donc pas d'ancre, est à son tour initialisé ce qui permet son expansion. Dans un second temps, une fois ce sous-arbre reconnu, on valide l'adjonction en considérant l'expansion de la seconde ancre de l'arbre auxiliaire.
3. Le troisième cas enfin, table 1.7, concerne les arbres auxiliaires englobant qui n'ont pas d'ancre d'un des deux côtés de l'épine dorsale de l'arbre. Dans ce cas de figure, il est possible que le nœud pied ne soit pas initialisé complètement par expansion des ancres de l'arbre auxiliaire. Il faut alors initialiser la position du nœud pied du côté où il n'y pas d'ancre, à tous les indices possibles entre 1 et $n-1$. La suite du processus est alors classique et correspond à l'adjonction englobante des cas 1 ou 2, suivant la distribution des ancres de l'arbre élémentaire sur lequel portera l'adjonction.

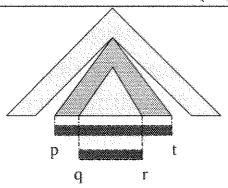
Nous avons caractérisé toutes les possibilités d'expansion des ancres des arbres auxiliaires englobants quelle que soit leur distribution par rapport au nœud pied.

Règle pour l'adjonction englobante cas 1



Initialisation du pied cas 1 :

$$\frac{(p,q,\sigma_L,\sigma_R) \qquad (q,r,\sigma_L{}',\sigma_R')}{(p,r,\sigma_L,next(\Gamma_R),q,r,N_0)} \quad \begin{array}{l}(\ \exists N_0/(N_0 \in \Gamma'_L) \wedge (N_0 \in \Gamma'_R) \quad (\sigma_L \wedge \sigma_R) \in \Delta \\ \Delta \in A \quad tail(\Gamma_R) = N* \quad head(\Gamma_L) = N^{\triangle} \quad)\end{array}$$

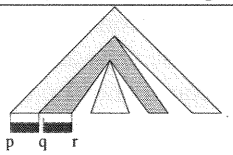


Adjonction englobante cas 1 :

$$\frac{(p,t,\sigma_L,\sigma_R,q,r,N_0) \qquad (q,r,\sigma_L{}',\sigma_R',u,v)}{(p,t,previous(N_0),next(N_0),u,v)} \quad (\ \exists N_0/(N_0 \in \Gamma'_L) \wedge (N_0 \in \Gamma'_R) \quad)$$

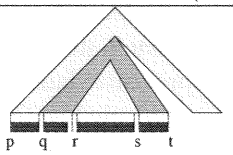
TAB. 1.5 – *Règles d'inférence pour le cas 1 de l'adjonction englobante.*

Règle pour l'adjonction englobante cas 2



Initialisation du pied cas 2 :

$$\frac{(p,q,\sigma_L,\sigma_R) \qquad (q,r,\sigma_L{}',\sigma_R')}{(r,r,next(N_0),next(N_0),N_0)} \quad \begin{array}{l}(\ \exists N_0/(N_0 \in \Gamma_R) \quad (\sigma_L{}' \wedge \sigma_R') \in \Delta \\ \Delta \in A \quad tail(\Gamma_R') = N* \quad head(\Gamma_L') = N^{\triangle} \quad)\end{array}$$



Adjonction englobante cas 2 :

$$\frac{(p,q,\sigma_L,\sigma_R) \qquad (q,t,\sigma_L{}',\sigma_R',r,s) \qquad (r,s,\sigma_L{}'',\sigma_R'',u,v,N_0)}{(p,t,\sigma_L,next(N_0),u,v)}$$

$$\text{tel que} \quad \begin{array}{l}(\ \exists N_0/(N_0 \in \Gamma_L{}'') \wedge (N_0 \in \Gamma_R{}'') \wedge (N_0 \in \Gamma_R) \\ tail(\Gamma_R') = N^{\triangle} \quad head(\Gamma_R') = N^{\triangle} \quad)\end{array}$$

TAB. 1.6 – *Règles d'inférence pour le cas 2 de l'adjonction englobante.*

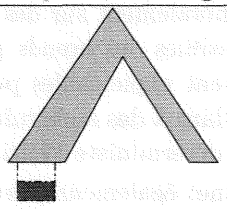
Les cas de figure 2 et 3 restent en pratique des cas rares, voir des cas d'école. On peut cependant concevoir des exemples linguistiquement fondés illustrant ces deux dernières configurations. Certains verbes à complétives peuvent être représentés à l'aide d'arbre auxiliaire englobant prédicatif, comme *forcer* dans l'exemple (38) ou *appeller*, exemple

(39), en considèrant les arbres de la figure 1.6[42].

(38) *Jean force à venir Marie.*

(39) *Jean appelle partir en vacances une perte de temps.*

Règle pour l'adjonction englobante cas 3



Initialisation du pied cas 3 :

$$\frac{(p,q,\sigma_L,\sigma_R)}{(p,r,\sigma_L,next(\Gamma_R),q,r)_{1\leq r\leq n-1}} \quad \left((\sigma_L \wedge \sigma_R) \in \Delta \quad \Delta \in A \quad tail(\Gamma_R) = N* \quad head(\Gamma_L) = N^{\Delta} \right)$$

TAB. 1.7 – *Règles d'inférence pour le cas 3 de l'adjonction englobante.*

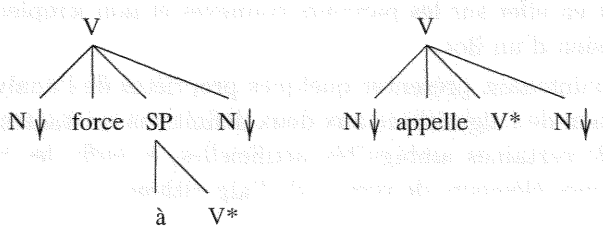


FIG. 1.6 – *Exemples d'arbres auxiliaires englobants relevant du cas 3.*

Nous pouvons également illustrer le troisième cas à l'aide d'un exemple abstrait, figure 1.7.

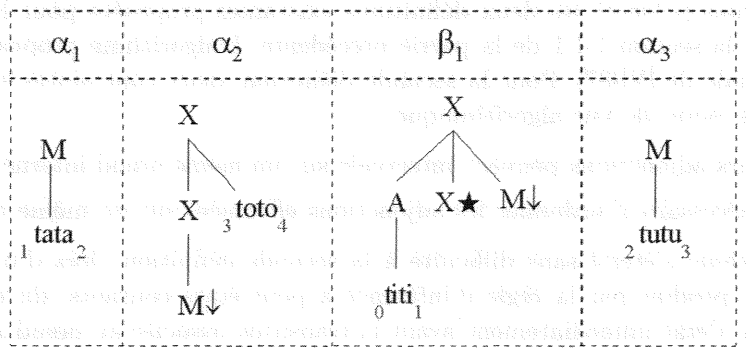


FIG. 1.7 – *Un exemple de configuration nécessitant une initialisation d'une des positions du nœud pied sur tous les indices possibles (exemple inspiré par Ariane Halber).*

42. Merci à Anne Abeillé pour ces deux exemples.

L'ensemble du processus d'analyse que nous venons de spécifier s'applique sur toute grammaire TAG lexicalisée. La lexicalisation est essentielle puisque l'analyse par connexité repose sur l'expansion d'îlots, donc initialement sur l'expansion des ancres et des co-ancres. Ces expansions permettent d'initialiser les positions des nœuds de substitution et des nœuds pieds. Ceci évite l'emploi de prédiction et assure une robustesse des solutions partielles qui, mise à part pour les *items* initialisés dans le cas 3 d'adjonction englobante, correspondent toujours à des îlots cohérents par rapport à la grammaire et la chaîne d'entrée. À l'opposé, les algorithmes existants s'appuient généralement sur des prédictions ou une initialisation indéterministe à tous les segments possibles des nœuds pieds, ce qui rend les indices de nombreux *items* incohérents, difficilement exploitables pour d'éventuelles réparations, et augmente le nombre d'*items* correspondant à des ambiguïtés artificielles. L'algorithme que nous proposons permet de rendre plus déterministe l'initialisation des nœuds pieds.

La notion de parcours connexe permet également de réduire le nombre moyen d'*items*. Grâce aux réductions de parcours connexes, tous les états des automates ne sont pas énumérés systématiquement. Par rapport à des algorithmes introduisant une règle de *non adjonction* pour l'avancement de la reconnaissance des arbres élémentaires, la *non-adjonction* est ici constamment implicite. Elle s'exprime en considérant les nœuds sur un parcours connexe, plutôt qu'un après l'autre. Il est ainsi possible de considérer une substitution d'un arbre initial alors que ses états d'expansion gauche et droite ne coïncident pas avec les états initial et final de l'automate représentant cet arbre. Les conditions d'application des règles portent en effet sur les parcours connexes et non simplement sur les deux états courants d'expansion d'un îlot.

Nous allons maintenant présenter quelques propriétés de l'analyse par connexité : tout d'abord l'adaptation de l'algorithme aux deux définitions existantes des dérivations LTAG, puis le blocage de certaines ambiguïtés artificielles, et enfin les résultats de complexité d'analyse et quelques éléments de preuve de l'algorithme.

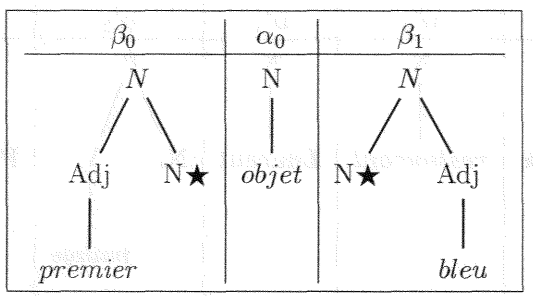
1.3 Propriétés

1.3.1 Définition des dérivations

Nous avons présenté les deux définitions existantes proposées pour les dérivations en LTAG dans la section 1.1.1 de la partie précédente. L'algorithme proposé respecte la définition initiale de [VS87]. Pour la seconde définition, deux contraintes sont à prendre en compte d'un point de vue algorithmique :

- plusieurs adjonctions peuvent intervenir sur un même nœud interne ou racine ;
- il est nécessaire d'ordonner les adjonctions effectuées sur un même nœud.

L'algorithme s'étend sans difficulté à la seconde définition : lors d'une adjonction, le nouvel item produit par la règle d'inférence a pour états courants, du côté où a eu lieu l'adjonction, l'état immédiatement avant la transition associée au nœud site d'adjonction. Il n'y donc plus une transition à suivre pour passer à l'état suivant après qu'une adjonction ait été évaluée à un état donné comme pour la définition de [VS87]. Pour illustrer ceci, nous donnons ci-dessous, table 1.8, l'exemple des règles d'adjonction gauche et droite adaptées à la nouvelle définition de [Schabes94]. Il suffit d'appliquer la même modification pour obtenir les règles pour l'adjonction englobante selon cette seconde définition.



TAB. 1.9 – *Grammaire LTAG pour l'analyse de (40)*

Adjonction gauche :

$$\frac{(p,q,\sigma_L,\sigma_R) \quad (q,r,\sigma_L{}',\sigma_R')}{(p,r,\mathbf{next(N)},\sigma_R')} \quad \begin{array}{l}(\exists (N^{\triangle} \vee N) \in \Gamma_L{}' \quad (\sigma_L \wedge \sigma_R) \in \Delta \quad \Delta \in A \\ head(\Gamma_L) = N^{\triangle} \quad tail(\Gamma_R) = N*_g \ )\end{array}$$

Adjonction droite :

$$\frac{(p,q,\sigma_L,\sigma_R) \quad (q,r,\sigma_L{}',\sigma_R')}{(p,r,\sigma_L,\mathbf{previous(N)})} \quad \begin{array}{l}(\exists (N^{\triangle} \vee N) \in \Gamma_R \quad (\sigma_L{}' \wedge \sigma_R') \in \Delta \quad \Delta \in A \\ head(\Gamma_L{}') = N*_d \quad tail(\Gamma_R{}') = N^{\triangle} \ )\end{array}$$

TAB. 1.8 – *Adjonction droite et gauche selon la définition des dérivation de [Schabes et al.94]*

1.3.2 Ambiguïtés artificielles

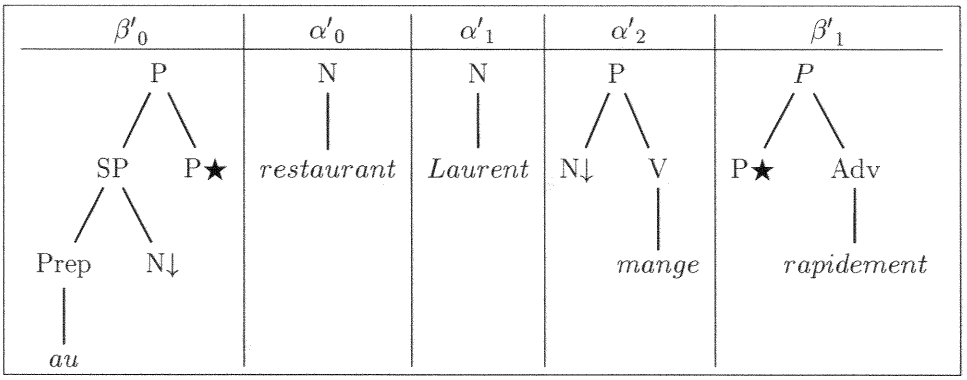
Les analyseurs existants amènent souvent des ambiguïtés artificielles liées au formalisme LTAG proprement dit et à des adjonctions en cascade. Précisons que ce problème n'a lieu que pour la définition des dérivations de [VS87]. Pour celle de [Schabes94], les adjonctions sont considérées comme multiples sur un même nœud.

En exploitant le fait que tout nœud susceptible d'être le site d'une adjonction est représenté par deux transitions dans l'automate, il est possible d'éviter très simplement ce type d'ambiguïté à l'aide de l'algorithme présenté. Il s'agit ici d'une optimisation de l'algorithme qui dépend de la façon dont est écrite la grammaire : on suppose pour l'appliquer que les arbres auxiliaires impliqués représentent des modifieurs dont l'ordre d'application n'est justement pas significatif. Ceci permet de ne pas avoir deux (ou plus) arbres de dérivation (facteur qui peut se propager exponentiellement lors de la suite de l'analyse) totalement équivalents du point de vue des dépendances sémantiques.

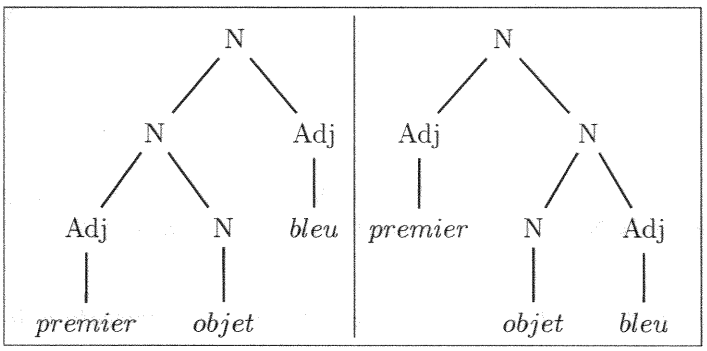
Commençons par illustrer ce phénomène fréquent. Considérons les deux énoncés suivants et la grammaire LTAG présentée table 1.10 (orthodoxe par rapport aux choix linguistiques habituels) :

(40) *premier objet bleu.*
(41) *Au restaurant Laurent mange rapidement.*

Dans les deux cas, il y a une adjonction en cascade soit au niveau de l'arbre élémentaire tête du groupe nominal pour le premier énoncé, c'est-à-dire α_0, soit au niveau de l'arbre élémentaire associé au verbe du second énoncé α'_2. Pour chaque énoncé on obtient deux



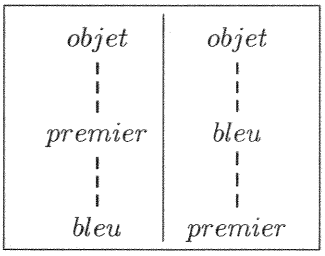
TAB. 1.10 – *Grammaire LTAG pour l'analyse de (41)*



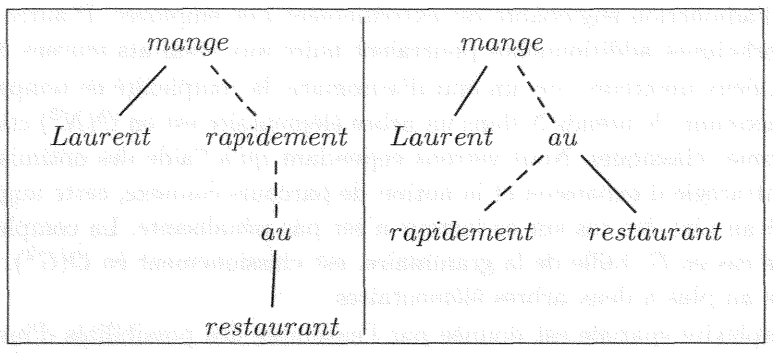
TAB. 1.11 – *Arbres dérivés LTAG pour l'analyse de (40)*

arbres de dérivations, voir figures 1.11, 1.12 et 1.13, sensibles à l'ordre dans lequel les adjonctions ont été réalisées.

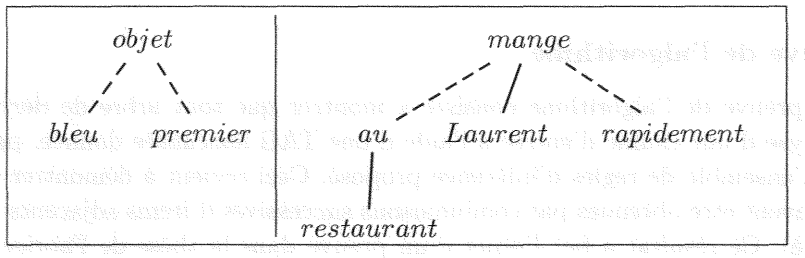
La solution proposée ici est d'exploiter le dédoublement des nœuds internes et racines issus de la linéarisation. Puisque l'ordre d'adjonction n'est en fait pas significatif, nous autorisons la prise en compte à ce niveau d'une adjonction droite et d'une adjonction gauche sur un même nœud, une première fois avec la première occurence de la catégorie associée au niveau des transitions de l'automate (adjonction gauche) et une deuxième fois avec la seconde occurence (adjonction droite). C'est en fait équivalent à une adjonction du



TAB. 1.12 – *Arbres de dérivation LTAG pour l'analyse de (40)*



TAB. 1.13 – *Arbres de dérivation LTAG pour l'analyse de (41)*



TAB. 1.14 – *Arbres de dérivation LTAG pour l'analyse de (40) et de (41) avec adjonction double gauche/droite autorisée*

premier modifieur sur le second (ou inversement) puis une adjonction englobante sur la tête syntaxique modifiée, ou à une première adjonction gauche sur la tête puis une adjonction droite sur le modifieur déjà adjoint (ou inversement). Un seul arbre de dérivation est alors fourni, voir figure 1.14, qui suppose en fait une adjonction double sur la tête syntaxique. Notons que cette technique reste cohérente avec la définition de [VS87], puisque l'ordre des dérivations n'est ici pas réellement significatif.

Ce problème de dérivations superflues est, bien sûr, lié au niveau suplémentaire introduit lors d'une adjonction et à la contrainte de se limiter à une unique adjonction par nœud. Par exemple, le formalisme TFG, dont l'opération de furcation est *plane*, ne présente pas ce problème.

1.3.3 Complexité

En étant proche des schémas d'analyse de [Sikkel97] pour la description du processus d'analyse, les résultats de complexité s'expriment sans difficulté en examinant le nombre de variables libres dans les différentes règles d'inférence. La règle la plus complexe est la règle d'adjonction englobante pour le cas 2. Elle compte sept indices libres de position, correspondant à une complexité en temps en $\mathcal{O}(n^7)$. Ce résultat peut être diminué à $\mathcal{O}(n^6)$ en utilisant les techniques présentées dans [Sikkel et al.93], comme l'a précédemment noté [vN94] pour l'adaptation de l'algorithme Head Corner au LTAG. En effet, on remarque dans la règle d'adjonction englobante cas 2 que l'indice q ne joue pas de rôle et a déjà été évalué dans l'initialisation du pied cas 2. Notons que nous ne nous sommes pas intéressé à cette optimisation qui se fonde sur l'utilisation de tables intermédiaires. Ceci, d'une part,

parce que l'adjonction englobante est extrêmement peu employée. D'autre part, parce que de telles techniques additionnelles pourraient nuire aux résultats moyens d'analyses.

Gérant deux pointeurs vers un état d'automate, la complexité en temps par rapport au nombre maximum de nœuds N dans un arbre élémentaire est en $\mathcal{O}(N^2)$ contre $\mathcal{O}(N)$ pour les algorithmes classiques. Nous verrons cependant qu'à l'aide des optimisations permises par notre stratégie d'expansion et la notion de parcours connexe, cette augmentation de la complexité au pire des cas sur ce facteur n'est pas pénalisante. La complexité temporelle au pire des cas en G, taille de la grammaire, est classiquement en $\mathcal{O}(G^2)$, puisque chaque règle réfère au plus à deux arbres élémentaires.

La complexité spatiale est donnée par l'ensemble des possibilités d'instanciation d'un *item*, soit une complexité spatiale au pire des cas en $\mathcal{O}(NGn^4)$ pour les LTAG, en $\mathcal{O}(NGn^2)$ pour les TIG, les deux indices étant inutilisés.

1.3.4 Preuve de l'algorithme

Une preuve de l'algorithme consiste à montrer que tout arbre de dérivation solution de l'analyse d'une chaîne d'entrée à l'aide d'une TAG lexicalisée donnée, peut être obtenu grâce à l'ensemble de règles d'inférence proposé. Ceci revient à démontrer que ces dérivations peuvent être obtenues par combinaisons successives d'*items* adjacents (contraintes de connexité). Ce résultat a fait l'objet d'un preuve dans la thèse de Fabrice Issac [Issac97] (voir pages 84 à 85).

Pour montrer que les règles d'inférence introduites couvrent bien toutes les possibilités de distribution des ancres, nous partons de l'arbre dérivation solution d'une analyse. Étant donné un arbre de dérivation, il faut pouvoir montrer que celui-ci peut être construit de façon itérative à l'aide des différentes opérations introduites. Chaque nœud d'un arbre de dérivation peut être décoré de la façon suivante :

- par deux indices correspondants à la position de la sous-chaîne dominée, pour tout arbre, excepté les arbres auxiliaires englobants ;
- par quatre indices repérant les deux sous-chaînes non contiguës dominées par l'arbre, pour tout arbre auxiliaire englobant.

Étant donné un nœud ν d'un arbre de dérivation, nous noterons cette décoration $\nu_{[i,j][k,l]}$ si ν représente un arbre auxiliaire englobant, $\nu_{[i,l]}$ sinon.

Soit un nœud père de l'arbre de dérivation ν, nous notons ses n fils ν_0,...,ν_{n-1}. La complétude des règles d'inférence revient à montrer qu'étant donnés les n *items* ν_0,...ν_{n-1} et le ou les *items* relatifs à l'initalisation de ν notée I_ν, il est possible de retrouver la décoration correcte de ν et l'ensemble des liens père/fils tout en combinant les *items* de façon adjacente. Si ceci est vraie pour ν, cela le sera itérativement pour l'ensemble de l'arbre de dérivation.

Pour le cas trivial où il n'y a pas de nœud fils, c'est-à-dire $n = 0$, l'arbre représenté par ν n'a pas de nœud de substitution (sinon il admettrait au moins un fils dans l'arbre de dérivation). La décoration correcte est donnée par l'initialisation de l'arbre : deux segments non contigus pour les arbres auxiliaires englobants, un seul pour les autres arbres élémentaires.

Dans les configurations non triviales, deux cas de figures peuvent se présenter :

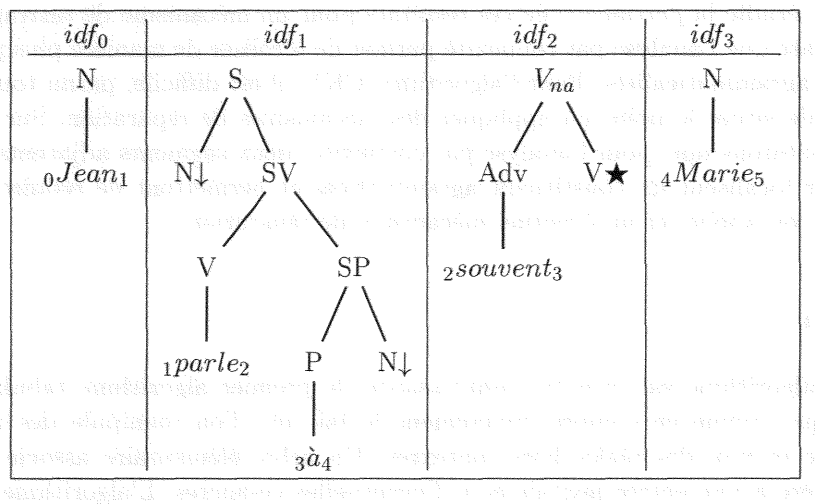
- ν n'est pas un arbre englobant : I_ν fournit donc un intervalle $[p,q]$, où $i \leq p$ et $q \leq l$. La preuve fournie par [Issac97] montre qu'il est possible d'étendre successivement par combinaison des *items* adjacents relatifs à ν_0,...ν_{n-1} jusqu'à création d'un *item*

représentant $\nu_{[i,l]}$. Cette extension est rendue possible par l'ensemble des règles de substitution, d'adjonction droite, gauche et la réduction des co-ancres.

- ν est lui-même un arbre englobant : dans ce cas I_ν peut fournir deux intervalles [p,q] et [p',q'], relatifs à l'initialisation des ancres respectivement à droite et à gauche de l'épine dorsale de l'arbre représenté par ν. Si ces indices sont tous définis, alors au moins une ancre est présente de chaque côté de l'épine dorsale, et à nouveau il est possible d'étendre ces deux intervalles jusqu'à création d'un *item* représentant $\nu_{[i,j][k,l]}$ à partir des *items* représentants $\nu_0,...\nu_{n-1}$, ce qui correspond aux cas d'adjonction englobante 1 et 2. Si, par contre, un des intervalles fournis par I_ν ne peut être spécifié, un des côtés de l'épine dorsale de l'arbre représenté par ν ne présente pas d'ancre. La solution dans ce cas de figure consiste à initialiser l'intervalle non spécifié successivement à tous les [r,r], où $1 \leq r \leq n-1$. Cette configuration d'analyse est traitée par le cas 3 des règles pour l'adjonction englobante. L'expansion des deux intervalles peut alors avoir lieu comme pour le cas précédent (cas 1 et 2), sachant qu'une seule de ces initialisations est correcte. Notons que les deux intervalles fournis par I_ν ne peuvent être tous deux non spécifiés puisque nous considérons que la grammaire LTAG est lexicalisée.

1.4 Intérêt des résultats fournis pour notre approche robuste

Nous présentons dans cette section l'intérêt des solutions fournies par l'algorithme d'analyse par connexité en vue d'effectuer des réparations d'analyses, ceci par rapport à ce que est fourni par les algorithmes classiques. Prenons l'exemple de l'énoncé *Jean parle souvent à Marie* avec la grammaire LTAG donnée figure 1.15. Cet énoncé est clairement agrammatical compte tenu de cette grammaire, puisque l'adverbe *souvent* est attendu avant le verbe. Par contre un analyseur robuste doit pouvoir, à notre sens, identifier les segments grammaticaux corrects, comme expliqué dans la partie précédente en 3.5, et les arbres de dérivations possibles pour ces segments.



TAB. 1.15 – *Grammaire LTAG pour l'analyse de Jean parle souvent à Marie .*

Nous comparons figure 1.8 les résultats fournis en fin d'analyse par un algorithme de type CKY généralisé et ceux fournis par une analyse par connexité. Nous supposons ici que les algorithmes utilisés procèdent à une expantion des îlots à partir de chaque ancre et

co-ancre. Nous constatons que, malgré l'agrammaticalité, une analyse par connexité a pu mener une réelle analyse bidirectionnelle, alors que l'analyse classique ne parvient jamais à remonter jusqu'au nœud SV de l'arbre élémentaire idf_1. Les segments grammaticaux fournis sur cet exemple sont plus étendus que pour les algorithmes classiques et correspondent aux objectifs fixés. Les arbres de dérivation partiels obtenus sont également plus riches comme le montre la figure 1.9.

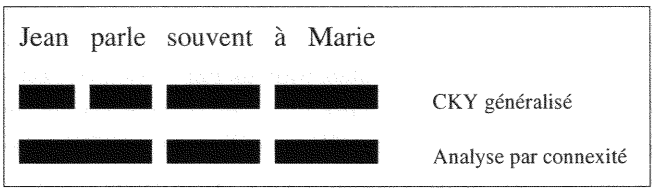


FIG. 1.8 – *Comparaison des segments grammaticaux fournis par l'algorithme CKY généralisé et l'analyse par connexité pour l'analyse de Jean parle souvent à Marie .*

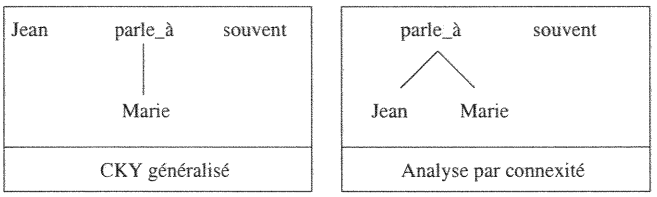


FIG. 1.9 – *Comparaison des arbres de dérivation partiels fournis par l'algorithme CKY généralisé et l'analyse par connexité pour l'analyse de Jean parle souvent à Marie .*

Si on étudie la pertinence de ces résultats pour un mécanisme de rattrapage ultérieur, on constate que l'analyse par connexité permet de localiser de manière plus précise la position des agrammaticalités. Pour l'algorithme CKY, il est difficile, parmi tous les segments fournis, de situer le point où appliquer des mécanismes de réparation. Sur l'exemple 1.8, nous constatons que, pour l'analyse par connexité, deux segments adjacents impossibles à rattacher localisent un constituant agrammatical et permettent de réduire le nombre de positions où appliquer un éventuel mécanisme de réparation.

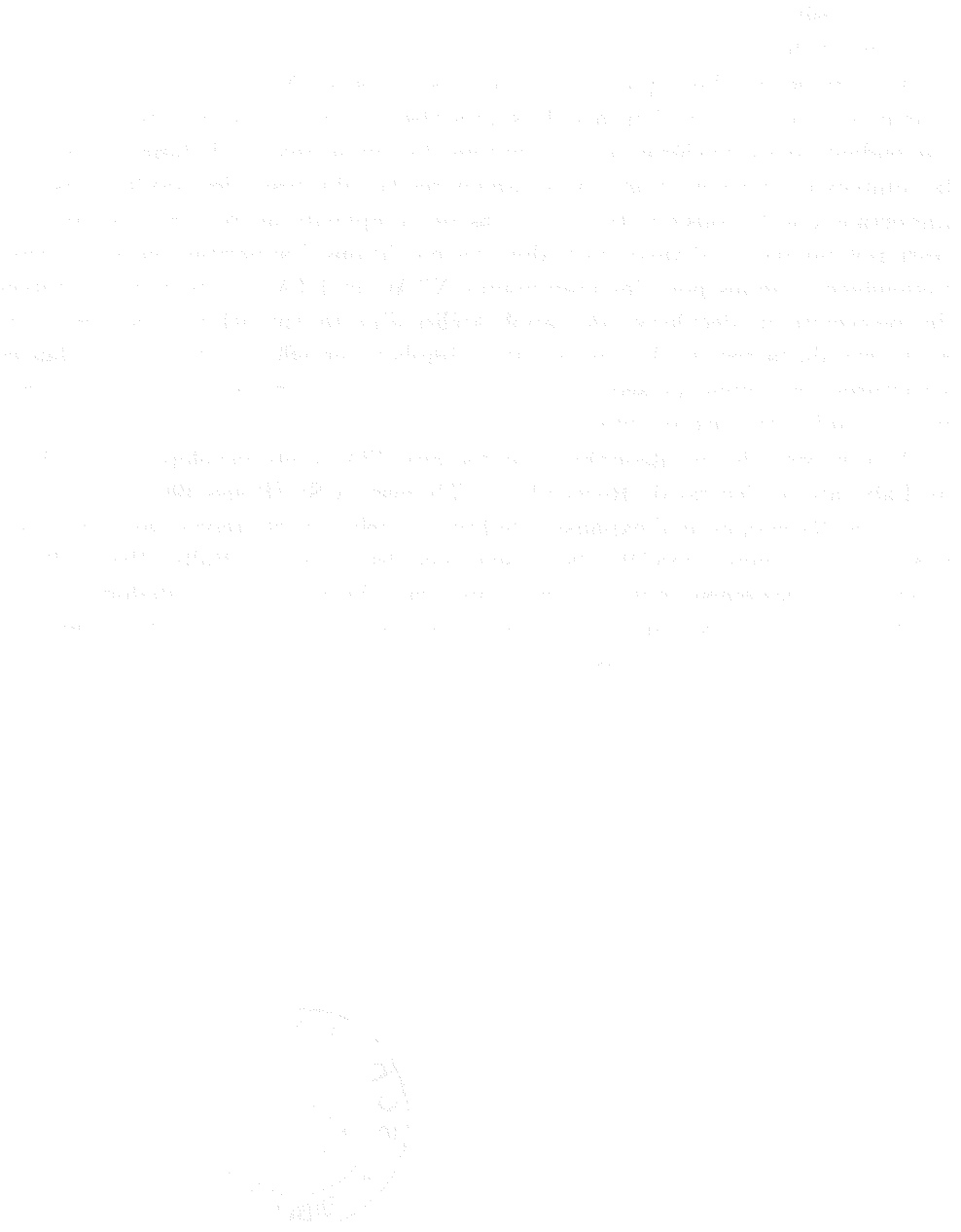
1.5 Bilan

Cet algorithme est, à notre connaissance, le premier algorithme tabulaire dédié aux LTAG qui prenne en compte directement le fait que l'on manipule des arbres partiels d'analyse et non des règles hors contextes. Un arbre élémentaire associe une structure particulière à une entrée lexicale et à d'éventuelles co-ancres. L'algorithme exploite cette structure pour obtenir des résultats partiels d'analyse plus étendus que ceux obtenus par les algorithmes directement issus de l'analyse de grammaires hors contextes, tout en gardant les avantages de techniques tabulaires d'analyse. Le résultat fourni en fin d'analyse est l'ensemble des segments grammaticaux de la phrase, ce qui correspond, à notre sens, à un résultat mieux adapté à l'analyse robuste de la langue. En quelque sorte, cet algorithme

tente de réaliser un compromis entre analyse superficielle par segment et analyse complète d'un énoncé fortement contrainte et guidée.

La complexité théorique au pire des cas est en $\mathcal{O}(N^2G^2n^6)$, sur ce plan l'algorithme n'améliore donc pas les résultats classiques observés pour ce style d'algorithmes tabulaires. Cependant nous considérons qu'en pratique, lorsqu'on analyse du langage naturel, diminuer la complexité moyenne d'analyse et améliorer la robustesse des résultats partiels est plus important que la considération d'un cas de complexité au pire des cas, qui, d'une part, reste polynomial et, d'autre part, dont les conditions d'apparition ne sont presque jamais rassemblées (jamais pour les grammaires XTAG et FTAG). Les deux véritables facteurs de complexité qui décideront de l'applicabilité d'un tel algorithme se situent premièrement au niveau du facteur G. La lexicalisation implique en effet de fortes redondances de sous-structures communes. Le second facteur de complexité important se situe au niveau de la phase d'unification exponentielle.

Une version de cet algorithme limitée aux TFG a été développée par David Roussel au Laboratoire Central de Recherche de Thomson CSF [Roussel99]. Notons aussi que les principes développés ici d'expansion bidirectionnelle ont été repris dans l'analyse par *chart* mise en œuvre dans le modèle d'analyse probabiliste d'Ariane Halber [Halber98b]. Ces deux travaux ont également pour domaine applicatif les systèmes de dialogue oraux homme-machine. Ceci montre l'intérêt des techniques proposées pour des domaines où pèsent de fortes contraintes de robustesse.



Chapitre 2

Développement des résultats

La spécification de l'algorithme présenté au chapitre précédent supposait l'utilisation d'un *chart* pour la mise en œuvre des tabulations. En fait, la réalisation de cette structure de données n'est pas immédiate et d'elle dépendent les performances de l'algorithme final. Nous avons évoqué l'intérêt de l'utilisation d'un *chart* et son principe de fonctionnement. Nous allons maintenant nous intéresser aux particularités et aux optimisations introduites afin d'assurer la cohérence globale des solutions partielles et afin de bloquer les dérivations redondantes inévitablement induites par le fonctionnement bidirectionnel de l'analyse. Nous présenterons ensuite l'étape d'extraction des dérivations de la forêt partagée d'analyse. Nous exposerons comment nous avons pris en compte les unifications du formalisme LTAG dans l'analyse. Ces unifications ne vont pas sans soulever un certain nombre de problèmes dont nous discuterons en fin de chapitre.

2.1 Contrôle et agenda

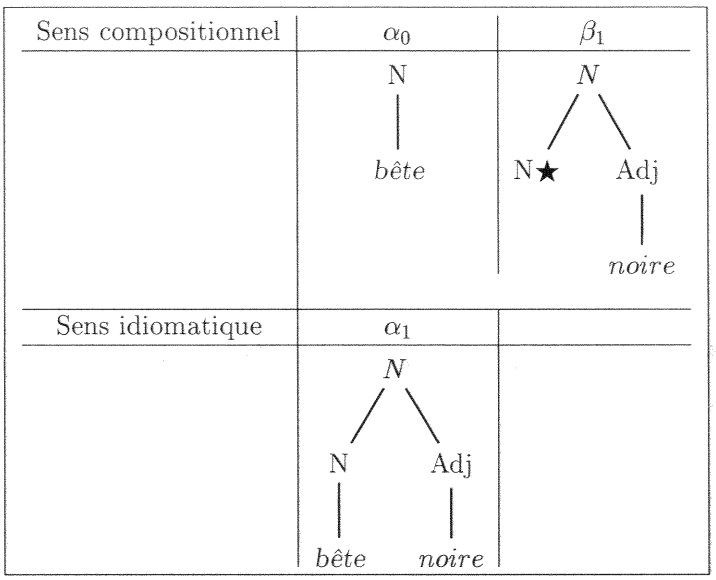
2.1.1 Analyse par chart

De manière classique aux analyses tabulaires, les *items* créés lors de l'analyse par connexité sont stockés dans une structure de données de type *chart*. Aucun *item* déjà tabulé ne peut à nouveau être introduit dans le *chart*. Afin de prévenir l'ajout d'un *item* qui existerait dans le *chart*, nous associons à chaque *item* un historique. Cet historique contient l'ensemble des *items* ayant fait l'objet d'une combinaison et peut lui-même prendre la forme d'un tableau pour assurer des comparaisons rapides entre opérandes concurrents et *items* interdits. En pratique, les *items* ne sont pas recopiés dans l'historique, mais simplement référés sous forme de pointeurs.

2.1.2 Contrôle et blocage des dérivations superflues

Cohérence lexicale globale des *items*

Cet historique est également employé pour établir une mutuelle exclusion entre des *items* correspondant à des hypothèses lexicales concurrentes. Ceci est particulièrement important pour assurer la cohérence des résultats partiels. Un *item* doit correspondre à une hypothèse sûre de segment grammatical. Or durant l'analyse, simplement en se basant sur les contraintes de combinaison présentées dans le chapitre précédent, une ancre peut venir se combiner à une co-ancre d'un arbre concurrent. Le segment résultant n'est pas valide



TAB. 2.1 – *Grammaire LTAG pour l'analyse de bête noire*

et l'*item* correspondant restera un *item* surgénéré sans suite car incohérent avec la fin de la reconnaissance des deux arbres. Pour illustrer ces productions d'ambiguïtés artificielles, prenons l'exemple de l'énoncé *bête noire* dont on peut distinguer le sens compositionnel (un animal de couleur noire) et le sens semi-figé (une personne peu appréciée). En considérant le principe de minimalité sémantique introduit en 1.2.3 de la partie précédente, nous aurons pour la grammaire LTAG deux constructions concurrentes comme le montre la table 2.1.

L'initialisation des trois arbres élémentaires aboutira à quatre *items*, un pour chaque ancre que nous pouvons noter de la façon suivante: $i^{\alpha_0}_{bête}$ pour l'ancre *bête* de l'arbre α_0, $i^{\alpha_1}_{bête}$ pour l'ancre *bête* de l'arbre α_1, et de même pour $i^{\beta_1}_{noire}$ et $i^{\alpha_1}_{noire}$. $i^{\alpha_0}_{bête}$ est en concurrence lexicale avec $i^{\alpha_1}_{bête}$, ce qui ne pose pas de problème puisque, compte tenu de leurs indices, leur combinaison ne sera jamais évaluée. Par contre $i^{\alpha_1}_{bête}$ et $i^{\beta_1}_{noire}$ sont également en concurrence lexicale, et une règle d'adjonction droite pourrait tout à fait amener ces deux *items* à se combiner. L'*item* résultat ne pourra pas poursuivre l'analyse, puisque la réduction de la co-ancre $i^{\alpha_1}_{noire}$ sera impossible. Cependant, si cet *item* est effectivement produit, il constitue tout de même une ambiguïté artificielle et un résultat partiel incohérent compte tenu de la grammaire.

Cette surgénération n'est pas d'une grande importance dans un algorithme où on se focalise sur les résultats complets. Cependant, dans notre cadre d'analyse robuste, il nous semble important d'assurer à tout moment la cohérence d'un *item* par rapport aux concurrences lexicales globales des arbres élémentaires. En plus de cette cohérence, une telle mutuelle exclusion permet de diminuer le nombre d'*items* du *chart*. Les inconvénients liés au blocage par mutuelle exclusion sont tout d'abord un historique de taille importante à gérer en termes de calcul et, comme nous allons le voir, une initialisation des ancres et co-ancres plus complexe.

La technique mise en œuvre emploie donc l'historique de chaque *item* de la façon suivante. Les *items* pouvant représenter (une fois combinés) des arbres dérivés dont les co-ancres se chevauchent, doivent être en mutuelle exclusion. Durant l'initialisation, on

ajoute en conséquence à l'historique de l'*item* associé à une ancre donnée notée a :

1. tous les *items* initialisant une ancre en position identique, située sur d'autres arbres élémentaires de cette grammaire (ensemble que nous appelons arbres concurrents de a) ;
2. tous les *items* initialisant les co-ancres de ces arbres élémentaires concurrents, ceci afin d'éviter qu'un *item* issu de a ne se combine ultérieurement avec un *item* lié à un arbre concurrent de a dont les indices peuvent être compatibles (cas illustré par l'exemple précédent *bête noire*) ;
3. tous les *items* initialisant des ancres ou co-ancres en position identique à une co-ancre de l'ancre a, ceci afin d'assurer la cohérence de l'ensemble de l'arbre élémentaire ancrant a. Ces *items* réfèrent à un ensemble d'arbres élémentaires dont l'arbre élémentaire ancrant a est également concurrent ;
4. enfin, tous les *items* initialisant des co-ancres des arbres précédents concurrents de l'arbre élémentaire ancrant a.

L'initialisation de cet historique peut être réalisée durant l'initialisation de l'ensemble des ancres. Chaque mot dans la phrase projette un ensemble d'arbres élémentaires concurrents. Les quatre ensembles d'*items* inclus pour l'initialisation de l'historique signifient simplement que l'ensemble des ancres et co-ancres de ces arbres sont concurrents et que, par extension, l'ensemble des *items* les initialisant doivent être en mutuelle exclusion.

Durant la production de nouveaux *items* en cours d'analyse, l'*item* produit hérite toujours de l'union des historiques des *items* arguments de la règle d'inférence. Ceci assure une propagation correcte des contraintes de mutuelle exclusion pour les nouveaux résultats partiels.

L'algorithme d'analyse par connexité associé à ces techniques de mutuelle exclusion d'*items* permet de satisfaire à tout moment de l'analyse :

- l'extension maximale des îlots compte tenu des contraintes d'adjacence et des contraintes structurelles locales sur les parcours connexes gauche et droit ;
- la correction globale des résultats partiels compte tenu de la concurrence des arbres élémentaires employés.

Blocage des dérivations superflues liées à la bidirectionnalité

En plus des ambiguïtés artificielles dues à des modifieurs gauche et droit en cascade que nous avons présentées dans la section 1.3.2, le risque d'une analyse ascendante bidirectionnelle est d'obtenir des dérivations redondantes. Dès les premiers tests, on constate en effet qu'il est courant d'obtenir une même dérivation suivant différents chemins dans la forêt de dérivation. Par exemple, étant donnés trois arbres élémentaires a, b et c, une même dérivation correspondant à abc peut être obtenue soit en associant d'abord a à b puis ab à c, soit en associant b à c puis a à bc. Il nous semble nécessaire de bloquer ces ambiguïtés artificielles si on désire mettre en œuvre un mécanisme d'expansion bidirectionnelle des îlots d'analyse. Ceci pour deux raisons : d'une part cette surgénération entraîne d'importants calculs superflus et d'autre part leur extraction de la forêt de dérivation est elle aussi coûteuse car exponentielle.

La solution retenue est classique aux analyses bidirectionnelles. Elle est inspirée de celle présentée dans ce contexte d'analyse bidirectionnelle initialement dans [Satta et al.89] et pour l'analyse bidirectionnelle de LTAG dans [Lavelli et al.91]. Cette technique se nomme *blocage par subsomption*. Un identifiant est associé à chaque *item* tabulé et une relation

d'ordre est établie sur les identifiants. Si a et b sont deux identifiants d'*item*, $b \prec a$ signifie que a subsume b ou encore que a est plus général que b. En pratique, il est inutile de placer a dans le *chart* si b y est déjà présent. Le faire entraînerait une redondance inutile et coûteuse des dérivations. Pour parvenir à ceci, les identifiants sont associés à des DAG (Graphes Orientés Acycliques) représentant le graphe de dérivation associé à l'*item*. Avant d'introduire un nouvel *item* dans le *chart*, on fait un test de subsomption avec les éventuels *items* tabulés en cette position. Si l'identifieur d'un nouvel *item* subsume un *item* déjà présent, alors il est inutile d'ajouter cet *item*. Dans ce dernier cas, on peut mettre à jour les historiques des *items* utilisés dans la règle d'inférence ayant produit l'*item* afin de ne plus tenter de générer cet *item* par la suite.

Cette technique correspond à une gestion ensembliste des dérivations : si une dérivation complète ou partielle figure déjà dans le *chart*, indépendamment de l'ordre des dérivations, celle-ci sera bloquée par le test de subsomption. À chaque nouvel *item* produit vérifiant le test de subsomption, on attribue un identifiant qui est le DAG résultant de l'union des identifiants des *items* l'ayant engendré.

2.1.3 Agenda

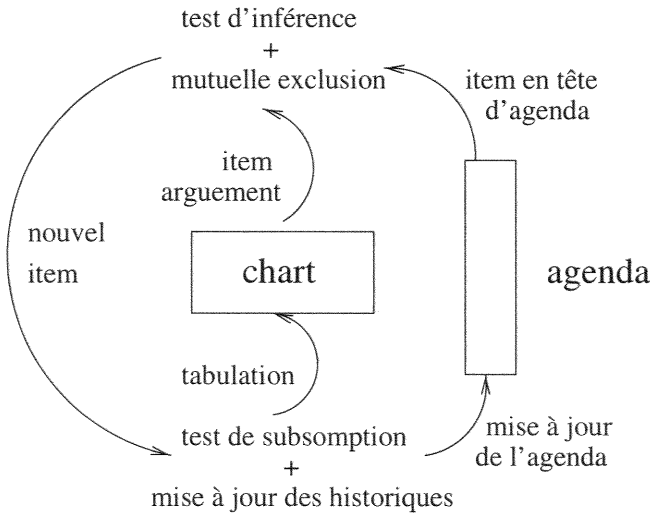


FIG. 2.1 – *Gestion des items d'analyse.*

L'analyse tabulaire permet de séparer la stratégie d'analyse, exprimée par les règles de composition d'*items*, du contrôle de l'analyse, exprimé dans la gestion d'un *agenda*. Dans le travail réalisé, nous n'avons pas étudié d'optimisation de l'agenda. Les grammaires LTAG étant lexicalisées, elles ne présentent pas de cas de récursion infinie et l'ordre dans lequel on sélectionne les *items* n'influe pas sur la terminaison de l'analyse. Nous avons réalisé nos tests avec un agenda correspondant à une pile *Dernier Entré-Premier Sorti*, de façon à favoriser les dernières hypothèses construites.

La figure 2.1 résume la gestion du *chart*, de l'agenda et des différentes techniques proposées. On sélectionne le premier *item* donné par l'agenda, on teste l'application des différentes règles d'inférence, en vérifiant pour les arguments sélectionnés dans le *chart* l'éventuelle mutuelle exclusion donnée par leur historique. Si un nouvel *item* est inféré, on

effectue un test de subsomption. Si ce dernier montre que l'*item* n'est pas redondant, il est tabulé dans le *chart* et ajouté à l'agenda.

Nous allons maintenant nous intéresser à la phase qui suit l'analyse par *chart*. Elle consiste en l'extraction des différents résultats et l'évaluation des unifications.

2.2 Forêt de dérivation et extraction des résultats

2.2.1 Principes

Afin de garder une trace des différentes dérivations réalisées durant l'analyse par *chart*, l'origine de chaque *item* est notée par les couples d'*items* l'ayant inféré. Le résultat d'une analyse est alors un graphe orienté d'*items* d'analyse, comme illustré par le schéma 2.2. L'ensemble des dérivations y sont présentes sous la forme d'une forêt partagée qui constitue une projection du graphe d'*items*. La forêt des arbres de dérivation peut ainsi être obtenue à partir de la forêt d'*items* simplement en ignorant les règles ne correspondant pas à des dérivations et en remontant toujours l'*item* associé à l'arbre sur lequel la dérivation a été effectuée. Dans notre algorithme d'analyse, les règles à ignorer sont la *réduction des co-ancres* et la *fin de parcours connexe*. Cette forêt de dérivations est obtenue et représentée polynomialement à l'issue de l'analyse par *chart*, bien que le nombre de dérivations soit au pire des cas exponentiel sur la longueur de la chaîne d'entrée.

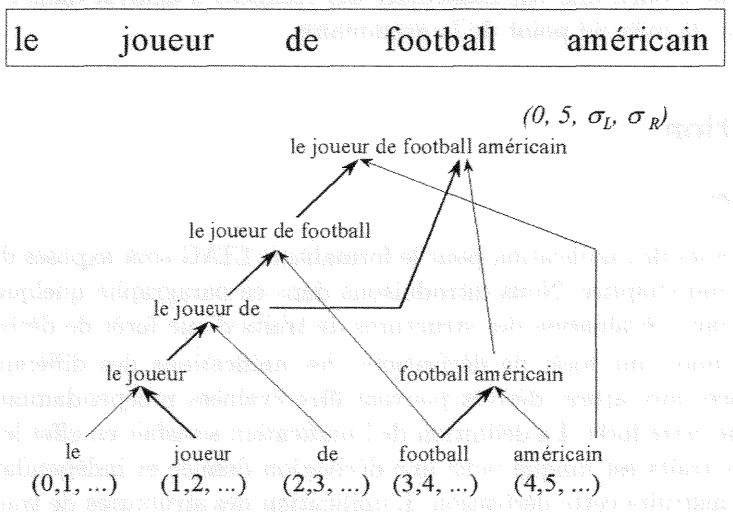


FIG. 2.2 – *Un exemple de graphe d'items résultant de l'analyse tabulaire de l'énoncé le joueur de football américain.*

L'extraction d'un résultat, c'est-à-dire d'un arbre de dérivation, ou d'un arbre dérivé, peut être obtenue linéairement. L'extraction de tous les résultats est par contre au pire des cas exponentielle puisqu'il faut énumérer l'ensemble des chemins possibles de la forêt. Dans le cas de TAG non entièrement lexicalisées, cette extraction peut même être infinie. Notons cependant que cette forêt partagée de dérivations constitue en soi un résultat intéressant sur lequel il est possible d'effectuer des traitements sémantiques ultérieurs exploitant également la factorisation des résultats.

Plutôt que d'extraire une forêt de dérivations, certains algorithmes tabulaires, comme celui de [Boullier96], créent une grammaire hors contexte non ambiguë dont les phrases

du langage sont l'ensemble des séquences de dérivations solutions de l'analyse. La grammaire solution encode alors l'ensemble de la forêt de dérivation. [Lang94] a montré que cette grammaire pouvait être vue comme l'intersection de la grammaire initiale et d'une grammaire régulière encodant simplement la phrase, ou un treillis de mots, à analyser. La grammaire capable de générer la forêt de dérivations peut alors être vue comme une spécialisation de la grammaire initiale.

2.2.2 Coût de l'extraction

Le coût de l'extraction des résultats correspond à un parcours complet de la forêt partagée à partir des *items* solutions. Ce coût peut être très important, ce qui est encore plus vrai lorsqu'on extrait toutes les solutions partielles. On peut alors arriver à un temps d'extraction des résultats proche et même supérieur à celui de l'analyse.

Ce coût ne nous semble cependant pas préjudiciable car, d'une façon pratique, si on souhaite exploiter cette structure, deux solutions sont envisageables :

- soit extraire un sous-ensemble des solutions en utilisant des contraintes d'autres niveaux linguistiques ou un algorithme de type A* ;
- soit garder la forêt sous sa forme partagée pour un traitement ultérieur.

L'extraction de l'ensemble des arbres de dérivation et des arbres dérivés réalisée a, avant tout, pour but d'offrir une vue immédiate des résultats d'analyse dans une phase de développement et de mise au point de la grammaire.

2.3 Unification

2.3.1 Principes

Les principes de l'unification pour le formalisme LTAG sont exposés dans le paragraphe 1.1.3 du second chapitre. Nous introduisons dans ce paragraphe quelques principes généraux concernant l'évaluation des structures de traits d'une forêt de dérivations partagées.

Étant donnée une forêt de dérivations, les unifications des différentes structures de traits associées aux arbres dérivés peuvent être évaluées indépendamment d'un parcours particulier de cette forêt. La définition de l'unification satisfait en effet le principe suivant : la valeur des traits est unique pour une dérivation donnée et indépendante de la manière dont a été construite cette dérivation. L'unification des structures de traits pour les LTAG est au pire des cas exponentielle dans la mesure où le nombre de résultats différents est lié au nombre de dérivations.

Dans la définition originelle des LTAG avec unification, les traits sont distribués au niveau de chaque nœud des arbres élémentaires. Il est plus simple de considérer qu'à chaque arbre élémentaire est associée une structure de traits globale du type :

$$[@arbre].[@noeud].[top|bottom].[attribut].[valeur\_atomique]$$

où $@arbre$ désigne le nom de l'arbre et $@noeud$ l'adresse de Gorn du nœud[43].

43. Cette adresse correspond à une séquence d'entiers positifs définie par induction de la façon suivante : la séquence vide notée 0 est l'adresse du nœud racine de l'arbre et p.k est l'adresse du k-ième fils du nœud d'adresse p.

Dans le cas d'organisations de grammaire LTAG par schèmes, qui est l'organisation classique car elle permet d'éviter une explosion du nombre d'arbres dans le lexique syntaxique, la structure de traits associée à α ne comporte que des traits de nature syntaxique. Les traits lexicaux sont projetés dans l'arbre après instanciation des ancres par unification. Le terme associé à une dérivation qu'il nous faut calculer peut être vu comme l'unification successive des DAG associés à chacun des arbres élémentaires, puis l'unification des traits lexicaux qui viennent partager habituellement la valeur du trait *bottom* des nœuds pères des ancres. Cependant des équations de traits supplémentaires sont introduites de façon non monotone lors de chaque opération de dérivation. La non-monotonie dans ce contexte est le fait que des traits associés à un nœud avant une opération, peuvent ne plus l'être après. C'est par exemple le cas du nœud site d'adjonction qui voit ses structures de traits réparties par unification dans les nœuds pied et racine de l'arbre adjoint. Tant que toutes ces équations liées aux dérivations n'ont pas été introduites pour un arbre élémentaire donné, il n'est pas possible de commencer l'unification des structures *top* et *bottom* des nœuds de cet arbre.

Ariane Halber a montré qu'en pratique les unifications liées aux opérations d'adjonction et de substitution ne peuvent pas faire tomber d'hypothèses. En effet, afin que les valeurs des traits restent indépendantes de la façon dont sont menées les dérivations, les structures *bottom* des nœuds de substitution et pied, ainsi que les structures top des nœuds racine, doivent rester vides[44] [Halber99]. La vérification des contraintes exprimées par les structures de traits est donc, en pratique, reléguée à l'unification finale des structures *top* et *bottom* associées à chaque nœud.

2.3.2 Unification et forêt partagée

Une fois une forêt de dérivations construite par un analyseur tabulaire, l'étape suivante consiste à évaluer les différentes structures de traits résultant de ces dérivations. Le but est de minimiser le nombre d'unifications et le nombre de parcours de la forêt.

Trois stratégies sont envisageables pour l'évaluation des unifications en complément de l'analyse proposée :

- une stratégie ascendante en cours d'analyse : cette solution permettra de faire chuter au plus tôt des analyses pour lesquelles l'unification échoue, en tenant compte des traits lexicaux dès le début du processus. Cependant, les unifications s'effectuant pour toutes les hypothèses partielles, elles sont également très nombreuses.
- une stratégie descendante après analyse : une fois la forêt partagée d'*items* fournie par l'analyse syntaxique, une solution est de coupler le parcours descendant de cette forêt nécessaire à l'extraction des dérivations avec l'unification. Les unifications sont effectuées au fur et à mesure du dépilement des arbres de dérivations par la racine. Si une unification échoue pour le rattachement d'un arbre donné α par exemple, on peut arrêter immédiatement le parcours pour la sous-forêt de dérivations dominée par α. La raison est que si l'unification échoue, alors que les traits associés à α à valeur réentrante ne sont pas encore instanciés, elle échouera pour toutes les instanciations de ces traits. Le premier intérêt de cette stratégie est donc de limiter les unifications aux dérivations que l'on souhaite extraire (les dérivations complètes ou les plus étendues par exemple). Le second intérêt est que ce processus permet de faire tomber des sous-

44. Notons qu'en pratique pour la grammaire du français ce n'est pas toujours le cas. Certaines structures *bottom* d'arbres auxilaires contiennent des traits qui auront pour objectif de bloquer directement certaines adjonctions.

forêts de dérivations lorsqu'une unification échoue. Par contre, les traits lexicaux ne sont pas employés au plus tôt au cours des unifications successives.

- une stratégie ascendante après analyse : une fois la forêt de dérivations produite, l'idée est de mener des parcours de la forêt à partir des *items* d'instanciation, puis de remonter jusqu'aux racines des arbres de dérivations. On commence par une extraction descendante des dérivations qui nous intéressent sur la forêt d'*items* d'analyse, puis on procède à un parcours ascendant pour les unifications de cette sous-forêt. L'avantage de cette méthode est de considérer au plus tôt pour l'unification les traits lexicaux, qui sont bien souvent les plus prédictifs. Son inconvénient est un parcours supplémentaire complet de la forêt de dérivation.

La solution mise en œuvre est la seconde stratégie qui nous paraît être un compromis intéressant entre efficacité et complexité d'implantation. En fait, une unification en cours d'analyse peut se révéler plus intéressante car elle permet un élagage des *items* directement durant cette analyse. Cependant, les unifications s'effectuant sur toutes les hypothèses produites, elles peuvent se révéler également coûteuses, si bien qu'il est difficile, sans les avoir toutes expérimentées et comparées, de savoir laquelle des trois stratégies serait la plus performante.

2.3.3 Stratégie descendante après analyse

En pratique, l'algorithme suivant est utilisé sur la forêt d'*items*. Supposons qu'on arrive à $item_0$ suivant un parcours descendant. Supposons que $item_0$ ait été construit, entre autres, suite à une opération d'adjonction ayant composé deux *items* pour en produire un troisième : $item_1 + item_2 \rightarrow item_0$, sachant que l'adjonction a lieu sur un nœud de l'arbre lié à $item_2$. Il y a un nombre au pire exponentiel de DAG associés à $item_0$. Pour chacun de ces DAG, en supposant que le DAG courant soit noté DAG_0, nous procédons alors de la manière suivante :

1. nous créons DAG_2, un DAG associé à $item_2$, et l'initialisons à une copie de DAG_0 ;
2. nous créons une copie de DAG_1, le DAG de référence de l'arbre lié à $item_1$ (l'adjonction va amener des traits de l'arbre lié à $item_2$ à partager des valeurs des traits de l'arbre lié à $item_1$, l'objectif étant d'amener ce partage à être effectivement vérifié pour $item_2$ qui représentera l'arbre lié à $item_0$ après adjonction) ;
3. on crée DAG_{adj} qui indique quels traits doivent voir leurs valeurs partagées (*cf.* la définition des unifications pour l'adjonction) ;
4. on unifie ces trois DAG : $DAG_2 = DAG_2 \cup DAG_1 \cup DAG_{adj}$; si l'unification échoue on peut reprendre en 1 avec un autre DAG_0 et ne plus considérer dans la suite des unifications et des extractions de dérivations, toute la forêt d'*items* dominée par $item_1$;
5. on élimine de DAG_2 les traits associés à la valeur du nœud site de l'adjonction, qui ne servent plus à rien, puisque ce nœud est désormais représenté par les anciens nœuds racine et pied de l'arbre auxiliaire.

Lorsqu'on a épuisé toutes les DAG de $item_0$, les étapes 1 à 5 sont répétées pour tous les autres couples d'*items* ayant produit $item_0$. Ce même traitement est répété récursivement de façon descendante sur les *items* ayant engendré $item_0$ et qui ne sont pas tombés durant l'étape d'unification.

2.3.4 Synchronie DAG/LTAG

La question de l'amélioration des performances de cette phase d'unification reste encore un sujet de recherche qui n'a pas reçu, jusqu'à présent, de réponses satisfaisantes. Une solution est de modifier le formalisme, c'est là une des motivations de la définition des DTG qui permettent, elles, des unifications monotones. [Halber99] a mené une étude très intéressante sur la nature des unifications dans le formalisme LTAG. Elle distingue 4 types de DAG s'écrivant indépendamment les uns des autres : *ref* qui représente les équations de traits d'un schème, *init* ceux relevant de l'instanciation lexicale, *dev* correspondant aux unifications lors des opérations d'adjonction et de substitution et *fin* qui représente les unifications finales entre les structures *top* et *bottom*. Une dérivation est valide si tous ces DAG s'unifient. Ainsi, le DAG *dev* devient équivalent à l'arbre de dérivations, la conjonction des DAG *ref* et *init* aux arbres élémentaires et le DAG final $ref \cup init \cup dev \cup fin$ à l'arbre dérivé. Ces propriétés permettent de se rapporter uniquement à l'arbre de dérivation pour le calcul des unifications. Les deux premières des trois stratégies proposées se retrouvent alors dans cette formalisation qui prouve leur correction (voir le chapitre 5 de [Halber99]).

Nous proposons dans les paragraphes suivants deux solutions à certaines des limites présentées par l'unification dans le formalisme LTAG. La première concerne la possibilité d'exploiter les traits pour contraindre des rattachements en cours d'analyse. La seconde vise à partager durant le calcul des unifications des équations de traits communes à plusieurs arbres élémentaires concurrents. Ces deux solutions n'ont pas encore été validées en pratique.

2.4 Prédiction par les traits

2.4.1 Intérêt

Avec la définition standard de l'unification pour les LTAG, il n'est pas possible de se servir des structures de traits pour contraindre les arguments possibles d'une opération. La raison est liée à la dissociation des traits *top* et *bottom* qui peuvent héberger des traits contradictoires, car une opération d'adjonction peut très bien les déplacer séparément vers deux nœuds différents. En d'autres termes, il ne sera possible de filtrer par les traits liés à un nœud donné qu'une fois certain que plus aucune opération n'interviendra sur ce nœud. Bien entendu, il sera trop tard. Or, les traits constitueraient en pratique un potentiel très important pour sélectionner les structures possibles étant donné un préfixe reconnu, ce qui correspondrait à guider par la grammaire un système de reconnaissance de la parole. Cela permettrait aussi dans une analyse ascendante de réduire la combinatoire des solutions partielles en tenant compte de certains traits au plus tôt. Cependant, à cause d'un certain nombre de phénomènes linguistiques (par exemple l'adjonction des modaux), une telle dissociation des traits est nécessaire et ne peut être remise en question sans perdre un fort pouvoir descriptif.

2.4.2 Structure de traits centraux

Afin d'isoler les traits pouvant aider à la prédiction, nous proposons d'ajouter à chaque nœud d'arbres élémentaires une structure de traits centraux (trait *center*, par analogie aux traits *top* et bottom). Cette structure rassemblera les traits n'ayant pas besoin d'une dissociation en *top* et *bottom*, et du même coup, permettra de regrouper les traits pouvant filtrer d'éventuels arguments pour une opération sur ce nœud.

Chaque nœud reçoit donc trois structures de traits : *top* et *bottom* dont la définition ne change pas, et la structure *center*. La structure *center* peut contenir la catégorie du nœud en l'exprimant sous forme de trait. En effet, les dérivations LTAG sont monotones sur la catégorie, elles ne viennent jamais modifier la catégorie d'un nœud où elle s'applique. *center* est une structure de traits subsumant *top* et *bottom*. Autrement dit, *center* est une structure généralisant à la fois *top* et *bottom*. *center* représente les traits de *top* et *bottom* qui resteront invariants au cours des opérations d'adjonction. *center* s'unifie à tout moment avec *top* et *bottom* (ce qui n'implique pas que *top* s'unifie avec *bottom*).

Les mécanismes habituels de gestion des traits *top* et *bottom* restent inchangés. Lors d'une substitution, les structures *center* de la racine de l'arbre substitué et du nœud de substitution sont unifiées (*center* filtre donc les arguments possibles). Dans le cas d'une adjonction, la structure *center* du nœud site d'adjonction doit s'unifier avec la structure *center* des nœuds racine et pied de l'arbre auxiliaire.

En exprimant la catégorie du nœud courant en tant que trait dans la structure *center*, le filtrage par catégorie lors d'une opération devient un cas particulier du filtrage permis par la structure de traits *center*.

En fin d'analyse, les structures *top*, *bottom* et *center* doivent s'unifier pour chaque nœud.

2.4.3 Construction de la structure de traits centraux

Étant donné un trait, le choix de sa mise ou non dans la structure *center* peut être obtenu de deux façons :

- suite à la décision du linguiste : lors de l'écriture d'un arbre élémentaire donné, pour chaque nœud, le linguiste place les traits dans *center* lorsqu'il veut imposer que leurs valeurs restent invariantes au cours des opérations d'adjonction sur ce nœud ;
- par un test automatique : pour chaque attribut de trait et chaque catégorie possible, on vérifie sur l'ensemble des arbres auxiliaires de la grammaire si ce trait garde une valeur invariante au nœud racine et au nœud pied. Si c'est bien le cas, on est certain que les adjonctions sur les nœuds de catégorie correspondante ne viendront jamais modifier la valeur de ce trait, que l'on pourra donc placer dans la structure *center*.

En pratique, pour une grammaire à large couverture telle XTAG, on constate que toutes les valeurs des traits morphosyntaxiques (nombre, genre, etc.) et syntaxiques (mode, voix, etc.) sont susceptibles d'être modifiées suite à une adjonction à un nœud donné (voir par exemple le traitement des auxiliaires et des modaux dans [Abeille91b]). A moins d'interdir l'emploi de certains phénomènes comme les extractions avec modaux ou auxiliaires, qui sont très fréquents, une telle constatation signifie que les grammaires LTAG classiques ne pourront jamais être exploitées efficacement pour diriger, par exemple, un système de reconnaissance de la parole. La raison principale est expliquée dans [Carroll et al.99a] et a été évoquée en 1.3.2 de la deuxième partie de ce mémoire : la localisation des dépendances sémantiques mise en œuvre dans les grammaires LTAG complique l'unification des traits morphologiques et syntaxiques. Une grammaire ne localisant que les dépendances syntaxiques n'a besoin que d'une seule structure de trait par nœud et ne fait appel à la percolation de traits que pour de très rares cas [Carroll et al.99a]. Si l'objectif est de guider un système de reconnaissance de la parole, une grammaire localisant les dépendances syntaxiques peut être plus appropriée. Or, [Kasper et al.95] notamment montre qu'une grammaire LTAG ne peut localiser les dépendances syntaxiques pour les phénomènes d'extraction et les

verbes à montée. Une grammaire DTG, par contre, est apte à localiser aussi bien les dépendances syntaxiques que sémantiques. Ce formalisme nous semble donc constituer la seule réponse vraiment satisfaisante au coûteux problème d'unification en deux temps du formalisme LTAG. Mais bien entendu, en se limitant aux contraintes syntaxiques, l'analyse risque d'être surgénérative car les contraintes sémantiques captées habituellement par les grammaires LTAG restent à prendre en compte lors de traitements ultérieurs.

2.5 Factorisation des équations de traits

Une solution pour diminuer la phase d'unification est de faire partager aux dérivations leurs équations de traits communes. Ceci suppose qu'on soit capable de les identifier, chose que le système de Marie-Hélène Candito [Candito99] a montré possible grâce à l'identification des fonctions (sujet, objet, etc.) dans les structures élémentaires. L'accord sujet-verbe par exemple pourrait ainsi se partager au cours d'une analyse pour plusieurs structures concurrentes. L'idée est alors d'associer un unique terme à l'ensemble des dérivations et d'augmenter le partage du DAG correspondant. Ainsi par exemple l'unification d'accord entre le sujet et le verbe ne sera évaluée qu'une seule fois pour toutes les dérivations concurrentes dont les arbres élémentaires mettent en œuvre cette même équation.

Notons que le format d'encodage des arbres élémentaires proposé par [Bonhomme et al.99] prévoit effectivement qu'on puisse noter ces équations de traits partagés. Cependant, il reste encore à exploiter effectivement en cours d'analyse ces informations obtenues par exemple avec le système [Candito99].

Pouvoir factoriser les équations de traits permettrait sans aucun doute une amélioration notable des performances de l'analyseur, dans la mesure où :

- ces équations sont très redondantes à cause de la lexicalisation et de la localité étendue ;
- l'unification joue en pratique un rôle important dans les contraintes syntaxiques et les spécifications linguistiques de la grammaire.

Là encore, cette solution demande à être validée. Sa mise en œuvre, à la différence de la solution des traits *center* dépend de la grammaire et nécessite des connaissances sur la technique de génération de la grammaire employée.

Nous avons, dans ce chapitre, présenté les techniques de mise en œuvre de notre algorithme tabulaire. Nous allons maintenant nous intéresser aux premiers résultats d'analyse que ces différentes techniques ont permis.

Chapitre 3

Premières évaluations

3.1 L'implantation

3.1.1 Petit historique

Une première version de l'algorithme présenté dans [Lopez98b] a été réalisée en C++, sans l'unification. L'algorithme n'employait pas d'automate mais des listes pour représenter les structures linéarisées des arbres et les parcours connexes. D'autre part, l'algorithme n'employait pas de *chart*, mais une structure de pointeurs et d'historique proche de [Issac97] pour simuler les arbres dérivés tandis que les arbres de dérivations étaient construits de manière synchrone. Autrement dit, sans tabulation, cet algorithme était irrémédiablement exponentiel au pire des cas.

Cette première implantation n'est cependant pas un échec complet, car il nous a permis de prendre conscience de l'étendue et de la vraie nature des problèmes. L'un des choix qui s'est révélé relativement malencontreux est celui du langage de programmation. C++ est certes rapide, mais l'efficacité d'un programme tient avant tout à la qualité de l'algorithme. Par comparaison, Java permet surtout de développer un programme portable, de façon surtout plus rapide et plus sûre, tout en assurant l'essentiel des concepts de la programmation objet. Question performance, si les compilateurs les plus récents pour le langage Java, dit *just-in-time*, restent de l'ordre de quatre fois plus lents que les compilateurs C++, rien n'interdit de porter le code sur une machine quatre fois plus rapide ou d'attendre quelques mois l'arrivée d'une machine plus puissante. Pour clore cet apparté technique, il nous semble qu'un facteur de rapidité de quatre se révèle peu de chose face à la rapidité de développement et la fiabilité du code obtenu à l'aide du langage Java, d'autant plus quand le développement a pour cadre un travail de recherche[45].

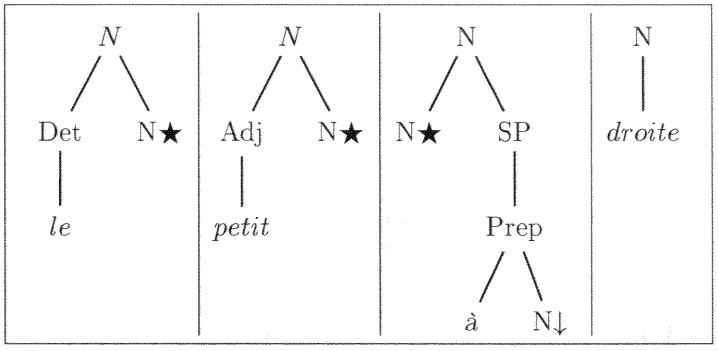
Aussi la deuxième implantation de l'algorithme a été réalisée en Java et intègre l'implantation complète d'un *chart* et des mécanismes d'unification. Nous allons décrire plus en détail son fonctionnement et ses caractéristiques dans la section suivante.

3.1.2 Descriptif de l'analyseur

Définition des dérivations

Cette seconde implantation permet de considérer la définition des dérivations selon [VS87] ou [Schabes94]. Un paramètre dépendant de la grammaire LTAG employée donne

45. Accessoirement Java intègre également des possibilités de mise en réseau intéressantes (*applet internet*).



TAB. 3.1 – *Grammaire LTAG pour l'analyse de (42)*

la défintion à appliquer durant l'analyse. Il s'agit, à notre connaissance, de la troisième implantation d'un système pouvant mettre en œuvre la définition de [Schabes94], avec le système de génération G-TAG [Danlos98] et l'analyseur probabiliste de [Schuler98], et du premier analyseur capable de faire fonctionner les deux définitions, ce choix étant un simple paramètre dépendant des choix de conception de la grammaire.

Pour des raisons de robustesse, les tests présentés dans le reste de cette thèse s'appuieront sur la définition de [VS87]. Pour s'en convaincre, nous pouvons examiner l'exemple suivant (42) d'ellipse sur une tête prédicative, en considérant la grammaire donnée table 3.1.

(42) *le petit à droite*

Avec la définition de [VS87], il est possible de combiner deux modifieurs se suivant sans la présence de la tête syntaxique modifiée, contrairement à la définition, moins souple, de [Schabes94]. Le résultat final présente, dans le premier cas, deux arbres partiels d'analyse, dans le second cas aucun rattachement n'est possible.

En s'appuyant sur cette définition, nous perdons certes au niveau de la qualité de représentation de l'arbre de dérivation, pour les raisons évoquées en 1.1.2 de la partie précédente, mais nous pouvons effectivement mettre en œuvre notre objectif d'analyse de l'ensemble des segments grammaticaux les plus étendus de l'énoncé.

Processus d'analyse

Compte tenu des choix de mise en œuvre présentés, l'analyse est le résultat de quatre étapes successives, de plus en plus coûteuses en temps de calcul.

Une première étape consiste à segmenter la chaîne à analyser en mots et à éliminer les différents parasites typographiques liés à la retranscription (majuscules, ponctuations, etc.). On applique ensuite un traitement pour éclater les *agglutinés* (appellés aussi *formes contractées*, par exemple *au* devient *à le*, *des* devient *de les*, *auxquels* devient *à lesquels*, etc.).

La seconde étape effectue l'initialisation des *items* d'analyse. Cette étape comporte une phase de sélection des arbres élémentaires projetés par les ancres conformément à un lexique syntaxique. Les *items* sont alors construits et leur historique est créé de façon à permettre une mutuelle exclusion de tous les *items* concurrents, comme expliqué dans le chapitre précédent en 2.1.2.

La troisième étape est la phase d'analyse proprement dite. Les îlots sont d'abord étendus simplement unidirectionnellement vers la droite. L'extension bidirectionnelle, plus coûteuse, est appliquée dans un deuxième temps aux analyses incomplètes. Ceci permet aux énoncés complets, qui ne nécessitent pas de bidirectionnalité, de s'effectuer plus rapidement. Cette étape s'arrête lorsque plus aucune règle ne peut s'appliquer sur le *chart*.

La quatrième étape enfin consiste à évaluer les unifications et à extraire les résultats d'analyse sous une forme immédiatement appréhendable, c'est-à-dire sous la forme d'un ensemble de couples (arbre dérivé, arbre de dérivation). La construction de la structure de traits associée à chaque dérivation et l'évaluation des unifications est effectuée de façon simultanée à l'extraction de ces deux arbres.

3.2 Tests sur un corpus de dialogues retranscrits

3.2.1 Le corpus GOCAD

Le corpus GOCAD[46] est un recueil d'interactions multimodales (voix et gestes) homme-machine, optenu suivant une manipulation de type Magicen d'Oz, dont nous avons présenté le principe (section 2.2.2 de la première partie). GOCAD est un logiciel de modélisation de surfaces naturelles complexes à partir de données hétérogènes, employé notamment en recherche pétrolière et en médecine. Huit sujets ont participé à l'expérimentation, chacun ayant quatre tâches à résoudre. La préparation et le protocole expérimental est décrit notamment dans [Chapelier96]. Plus qu'un système de commande vocale, le système simulé est capable d'apporter une assistance aux utilisateurs. Le résultat est un ensemble de dialogues complexes employant une richesse et une variabilité verbale très importante. Ce dernier point justifie le choix de ce corpus, car c'est pour ce type de système complexe que le travail mis en œuvre ici est destiné.

Le corpus GOCAD a donné lieu à différents travaux sur la modélisation des dialogues d'assistance [Chapelier96] et sur l'étude du codage de la référence dans un contexte multimodal [Bruneseaux et al.97].

Ce corpus en français correspond à de l'oral retranscrit. La tâche de retranscription de l'oral est particulièrement délicate car beaucoup d'informations, comme la prosodie, sont perdues. De même, la perception auditive de certains mots ou phénomènes est subjective. Les phénomènes oraux ont été annotés de façon similaire au codage proposé dans [Morel89], c'est-à-dire couvrant les hésitations, les interruptions et les pauses. Nous présentons tout d'abord quelques données quantitatives sur ce corpus. Le corpus a été découpé lors de sa retranscription en énoncés. Un tour de parole de l'utilisateur correspond à un énoncé. Un tour de parole de la machine est souvent constitué de plusieurs énoncés, ce qui explique les données du tableau 3.2.

nb. total d'énoncés	nb. total de tours de parole	nb. énoncés utilisateur
3200	1797	862

TAB. 3.2 – *Données générales sur le corpus GOCAD.*

Nous nous sommes interessé avant tout aux énoncés des utilisateurs, dont les données

46. Corpus codé selon la TEI (*Text Encoding Initiative*), disponible sur le serveur Silfide (`http://www.loria.fr/projets/Silfide/`)

générales sont présentées dans le tableau 3.3.

nb. total d'énoncés	nb. total de mots	nb. moyen de mots/énoncé
862	5535	6,42

TAB. 3.3 – *Données générales sur les énoncés des utilisateurs dans le corpus GOCAD.*

3.2.2 Lexique et grammaire du corpus GOCAD

Nous allons maintenant présenter le lexique et la grammaire développés pour l'analyse du corpus GOCAD. L'extraction du dictionnaire morpho-syntaxique correspondant au vocabulaire du corpus a été effectuée à l'aide du dictionnaire morphologique BDlex (320 000 entrées fléchies). Le tableau 3.4 donne des informations sur cette extraction.

nb. de formes fléchies	nb. d'ensembles lemme+traits	nb. de mot inconnu de BDlex
526	825	124

TAB. 3.4 – *Extraction morpho-syntaxique du lexique du corpus GOCAD depuis BDlex.*

Parmi les mots inconnus, qui se sont révélés très nombreux, la répartition et l'importance des causes d'échecs d'extraction sont résumées dans le tableau 3.5. A ces échecs s'ajoutent 17 mots présents dans tout dictionnaire de langue française mais omis dans BDlex comme *colorier* (seul *colorer* est présent), orbite, zoom, etc.

noms propres	fautes d'orthographe	interjections	nombres ou lettres	termes du domaine
26	15	6	29	4

TAB. 3.5 – *Taxonomie des échecs d'extraction morpho-syntaxique du lexique du corpus GOCAD depuis BDlex.*

Le lexique syntaxique associe à un couple (lemme, ensemble de traits) une liste de pointeurs vers des arbres élémentaires non-instanciés (les *schèmes*) avec les éventuelles co-ancres associées. Les arbres élémentaires sont dynamiquement instanciés au cours de l'analyse en sélectionnant les schèmes appropriés et en initialisation leurs ancres et leurs traits lexicaux (en pratique les traits morphologiques de l'ancre principale). Pour la conception des schèmes, nous avons suivi les principes de conception de la grammaire LTAG du français de TALaNa [Abeille91b] [Abeille et al.94], qui ont été présentés dans le chapitre 1 de la partie précédente. Ces principes sont dédiés à l'analyse de l'écrit et non de l'oral, les suivre est un parti-pris qui nous paraît important. En effet, nous avons choisi, dans la première partie de ce mémoire, une démarche partant de ressources grammaticales pour l'écrit en vue de les adapter à l'oral. Cette démarche nous permettra de voir jusqu'où peuvent aller de tels principes pour l'analyse de l'oral.

Le tableau 3.6 donne quelques informations sur le lexique syntaxique. Le nombre de *liens vers schèmes* donne la taille du lexique syntaxique si tous les arbres étaient prélexicalisés. Notons que le nombre de schèmes reste très modeste par rapport à la grammaire LTAG générale du français, qui compte maintenant de l'ordre de 5000 schèmes, mais les constructions syntaxiques du corpus se sont révélées relativement simples et répétitives.

Notons aussi que la grammaire réalisée ne comporte pas d'arbre auxiliaire englobant et correspond donc à une TIG.

nb. total de schèmes	nb. de liens vers schèmes
78	1776

TAB. 3.6 – *Grammaire LTAG pour le corpus GOCAD.*

Plus de détails sur la conception de ce lexique syntaxique LTAG pour le corpus GOCAD seront présentés dans la cinquième partie de ce mémoire consacrée aux outils d'assistance à la conception de grammaires LTAG spécialisées.

3.2.3 Résultats

Le tableau 3.7 présente les résultats de l'analyse des interventions de l'utilisateur du corpus GOCAD, soit 862 énoncés. Nous y donnons le pourcentage d'énoncés qui ont conduit à des analyses complètes. Une analyse complète est une dérivation couvrant l'ensemble des énoncés à analyser. Nous autorisons ici l'arbre dérivé d'une analyse complète à n'être pas entièrement saturé (nœuds de substitution non utilisés ou arbre auxiliaire non adjoints). Ce relâchement de contrainte a pour but d'assurer la couverture des différents cas d'ellipses. Nous reviendrons plus en détail sur ce point au chapitre 2 de la partie suivante. Il est clair que ces résultats d'analyses complètes n'ont qu'une valeur indicative et ne peuvent se substituer à une évaluation rigoureuse de précision fondée sur un corpus annoté. Cependant, l'annotation syntaxique du corpus GOCAD, qui constitue une tâche délicate, est une perspective proche.

Les temps d'analyse donnés au tableau 3.8 ont été obtenus sur une station Sun Ultra 1 et avec une machine virtuelle Java de 32Mo. Nous avons employé pour la phase d'analyse deux ensembles de règles d'inférence : celui présenté dans le premier chapitre de cette partie, correspondant à l'analyse par connexité, et celui décrivant un algorithme de type CKY généralisé prédictif (voir chapitre 3 de la deuxième partie). Ce dernier algorithme est, parmi les algorithmes classiques, l'un des plus proches du fonctionnement ascendant que nous proposons, tout en restant relativement simple à implanter. La gestion du *chart* et de l'agenda élémentaire était identique pour ces deux ensembles de règles et correspond à ce qui a été présenté au chapitre précédent. Les temps d'analyse donnés correspondent uniquement à l'étape d'analyse par *chart*, ils ne comprennent pas les phases de lexicalisation, d'unification et d'extraction des résultats, qui sont communes aux deux algorithmes d'analyse testés.

Un résultat partiel correspond ici à l'expansion maximale d'un îlot, autrement dit, dans le cadre de l'analyse par *chart*, à un *item* qui n'est origine d'aucun autre *item*. L'expansion moyenne par îlot est le nombre moyen de mots des résultats partiels. Cette expansion moyenne est donnée ici pour l'ensemble des analyses, qu'elles soient complètes (résultats recouvrant l'énoncé) ou non (résultats partiels).

On constate que les temps d'analyse sont un peu meilleurs pour l'analyse par connexité. L'augmentation du facteur de complexité en N, nombre maximum de nœuds dans un arbre élémentaire, et la délivrance des segments grammaticaux les plus étendus ont été compensées par la détermination des initialisations des nœuds pieds proposée par notre algorithme et par les optimisations permises par la notion de parcours connexes. Cette dernière, nous l'avons vu, permet de ne pas engendrer d'*items* pour toutes les positions

corpus	% analyses complètes	nombre moyen d'analyses par énoncé
Gocad	78,31	2,06

TAB. 3.7 – *Résultats globaux des analyses complètes du corpus Gocad.*

algorithme	temps moyen /énoncé (ms)	expansion moyenne /îlot
CKY généralisé prédictif	87	2.55
analyse par connexité	58	2.79

TAB. 3.8 – *Comparaison des temps d'analyse et des résultats partiels entre deux algorithmes d'analyse ascendante de LTAG.*

possibles sur les automates. Cependant il est clair que cet algorithme par connexité restera plus lent que des algorithmes mettant en œuvre des étapes efficaces de prédictions comme l'algorithme de Schabes ou l'adaptation du *Head Corner* au LTAG.

Le point le plus important, à notre sens, est que la longueur moyenne des résultats partiels est plus élevée à l'aide de nos techniques que pour un CKY généralisé. Cela confirme que l'analyse par connexité permet, dans certains cas, d'aller plus loin dans les analyses partielles que les algorithmes classiques.

Avant de nous intéresser aux configurations ayant mis en échec l'analyse et aux possibilités d'amélioration des résultats fournis, nous présentons une technique de factorisation des structures redondantes de la grammaire LTAG en vue d'accélérer le processus d'analyse. En effet, les systèmes de dialogue actuels peuvent employer un vocabulaire bien plus important que celui considéré ici pour le corpus GOCAD. Le système *Verbmobil*, par exemple, met en œuvre un vocabulaire pouvant aller au-delà de 5000 mots. En nous intéressant à ce type de techniques de précompilation, nous avons tenté de fournir une solution garantissant l'adéquation de nos techniques même pour de tels vocabulaires.

Chapitre 4

Techniques de compaction de grammaires lexicalisées d'arbres

4.1 Technique de compaction

4.1.1 Le coût de la lexicalisation et du domaine de localité étendu

La lexicalisation et le domaine de localité étendu, nous l'avons vu, soulèvent le problème de la multiplication de sous-structures identiques qui peut s'avérer très pénalisant. Avec une grammaire hors contexte, par exemple mise sous une forme de Chomsky, une même règle peut être utilisée pour toutes les analyses possibles contenant la sous-structure correspondante. Dans les Grammaires Lexicalisées d'Arbres, cette sous structure est dupliquée d'un arbre élémentaire à l'autre. En considérant les choix linguistiques classiques portant sur la conception de ce style de grammaires, les polystructures (par exemple les arbres élémentaires associés aux expressions *parler à...*, *parler à... de...*, *parler de... à...*) sont très communes. Les arbres élémentaires correspondant doivent, à notre sens, partager leurs sous-structures communes afin de ne pas coûter, lors de l'analyse, autant qu'en considérant tous les arbres comme tous indépendants. Le schéma 4.1 illustre un tel cas de figure, deux *items* différents réfèrent en fait à une même sous-structure partagée par deux arbres élémentaires différents.

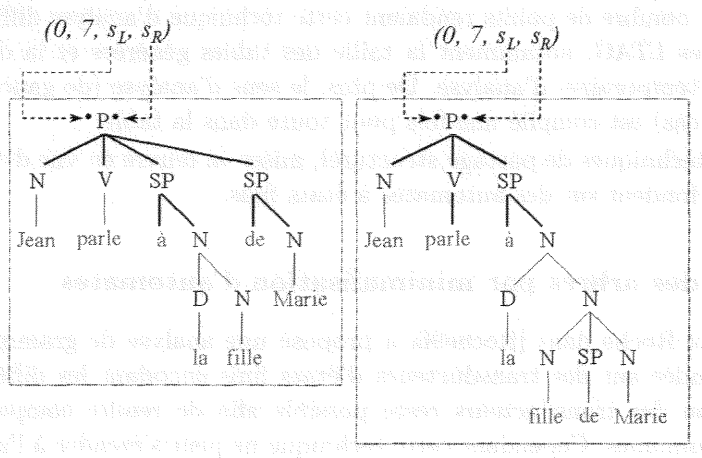


FIG. 4.1 – *Un exemple de résultat d'analyse sans partage structurel dans la grammaire LTAG.*

Tout comme une analyse par *chart* permet de réduire la complexité d'analyse en n en exploitant une factorisation des *items* selon leurs positions par rapport à la chaîne d'entrée et leurs positions par rapport à la structure analysée (règle, arbre ou automate), il est possible de réduire la complexité moyenne en G et N en utilisant une compaction des sous-structures communes, parce que chaque sous-structure commune implique des actions communes de l'analyseur. Une telle compaction permettra une diminution de la complexité moyenne d'analyse, la complexité au pire des cas, c'est à dire ici sans partage possible, restant inchangée.

Nous allons tout d'abord présenter les premières approches de compaction structurelle de grammaire d'arbres menées en vue d'accélérer l'analyse.

4.1.2 Une première approche : les TAG *schématiques*

Le formalisme des TAG *schématiques* a été introduit par David Weir [Weir88] dans le but d'aboutir à des compressions des descriptions syntaxiques. Suivant cet objectif, une grammaire TAG est étendue en considérant qu'il est possible pour chaque nœud interne ou racine de spécifier ses fils possibles à l'aide d'une expression régulière. La grammaire résultante est constituée d'un ensemble d'arbres élémentaires schématiques, chacun représentant lui même un ensemble d'arbres élémentaires classiques. Ce format se prête bien à la description des langues à ordre libre de mot comme l'allemand [Harbusch94].

Une grammaire TAG schématique consiste en pratique en un petit nombre d'arbres, mais en un nombre important d'expressions régulières. Pour l'analyse, cette représentation est intéressante dans la mesure où il est possible de factoriser la reconnaissance de certaines sous-parties communes d'arbres appartenant à une forêt d'arbres élémentaires codée par un même arbre schématique.

Cependant en pratique, la factorisation n'est pas optimale et la spécification par expressions règulières peut vite s'avérer surgénérative d'arbres.

4.1.3 Partage de préfixe dans l'analyse LR

La précompilation des actions d'analyse dans une table LR rend possible une stratégie de partage du préfixe commun de plusieurs règles de production. Nous avons cependant vu qu'un certain nombre de points rendaient cette technique d'analyse difficile à utiliser pour les grammaires LTAG, notamment la taille des tables générées et la difficulté de revenir sur des états temporaires d'analyse. De plus, le sens d'analyse (de gauche à droite dans la majorité des cas) est compilé une fois pour toute dans la table.

D'autres techniques de partage structurel, mises en œuvre en vue d'améliorer les temps d'analyse, se fondent sur des automates à états finis.

4.1.4 Partage des arbres par minimalisation d'automates

Emmanuel Roche dans [Roche96] a proposé une analyse de grammaire hors contexte lexicalisée fondée sur des transducteurs d'états finis encodant les différentes règles. Une minimalisation des transducteurs reste possible afin de rendre compte d'un partage de structures communes. Cependant cette technique ne peut s'étendre à l'analyse de langages légèrement dépendants du contexte, donc en particulier les langage d'arbres adjoints. De plus, cette technique n'emploie pas de tabulation ou de processus équivalents.

Roger Evans et David Weir ont proposé également de s'appuyer sur des automates à états finis minimalisés en vue de partager les différentes sous-structures communes [Evans et al.97] et d'exploiter ce partage dans une analyse tabulaire [Evans et al.98]. Leur objectif est l'analyse de texte tout-venant à l'aide d'une grammaire de type DTG de large couverture de l'anglais. Cette dernière compte actuellement de l'ordre de deux mille arbres élémentaires [Carroll et al.99b], ce qui constitue un véritable défi dans la mise en œuvre d'une analyse efficace. Une technique comme le *supertagging* [Srinivas97a] de désambiguïsation lexicale probabiliste, permet de diminuer le nombre de structures considérées durant l'analyse, mais élimine du même coup un nombre important d'ambiguïtés qu'il peut être important de conserver.

Cette technique est, en fait, proche du partage de préfixes des analyses LR. La reconnaissance d'un arbre élémentaire y est vue comme le parcours d'un automate à états finis dont les états correspondent à une traversée de l'arbre et les étiquettes de transition à des actions de l'analyseur. La figure 4.2 montre un arbre et l'automate correspondant. La reconnaissance ascendante à partir de l'ancre de cet arbre élémentaire nécessite l'exécution d'une séquence d'actions de l'analyseur en commençant par $\overrightarrow{NP}$ (trouver un NP sur la droite), puis un certains nombres d'actions $\overleftrightarrow{VP}$ (adjonction d'un arbre VP), une action $\overleftarrow{NP}$ (trouver un NP sur la gauche) et finalement un certain nombre d'action $\overleftrightarrow{S}$ (adjoindre un arbre dominé par un nœud S).

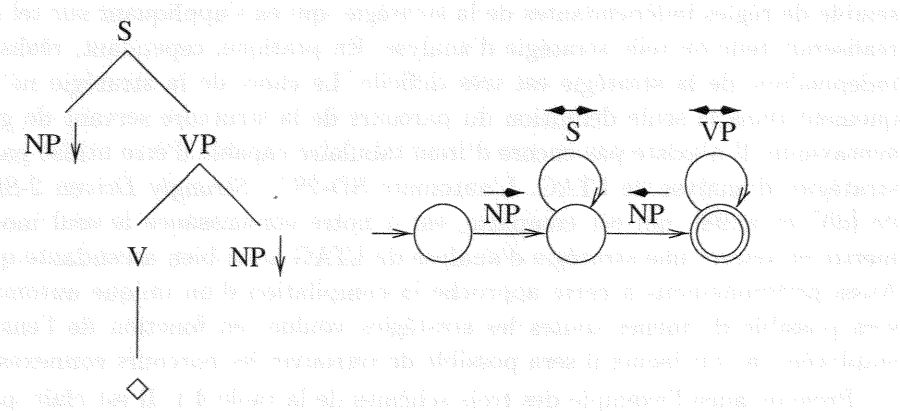
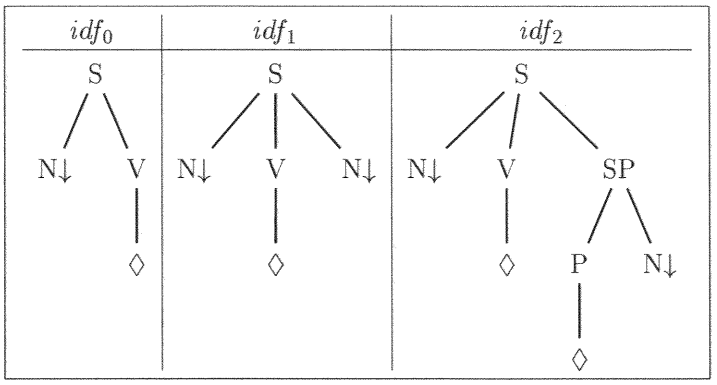


FIG. 4.2 – *Arbre élémentaire et l'automate correspondant dans [Evans et al.98].*

4.1.5 Notre approche

Comme nous nous sommes appuyé sur des techniques tabulaires pour notre algorithme d'analyse, nous avons décidé d'adapter la dernière approche à nos techniques d'analyse. Nous utilisons des techniques à états finis et de minimalisation similaires, mais nous employons des automates différents, beaucoup plus proches de ceux utilisés par Emmanuel Roche. La contrainte en effet était de pouvoir retrouver et suivre simplement les différents parcours connexes dans un automate représentant un arbre élémentaire. Ceci n'est pas possible dans ceux utilisés dans [Evans et al.97] qui encodent une reconnaissance ascendante particulière des arbres. Les automates que nous utilisons, nous l'avons vu, n'imposent pas une stratégie spécifique durant l'analyse, dans la mesure où ce sont les règles d'analyse qui définiront la marche de l'algorithme. Un ensemble particulier de règles appliqué à un même automate minimalisé amenera une stratégie d'analyse particulière, mais dans tous les cas



TAB. 4.1 – *Un exemple de trois schèmes redondants structurellement.*

une analyse avec partage des sous-structures.

L'objectif défini dans [Evans et al.97] est, à l'opposé, d'encoder un parcours particulier dans un automate, d'obtenir ainsi une série d'automates associée à un même arbre élémentaire mais définissant des parcours différents, par exemple ascendant de gauche à droite, à partir d'une ancre particulère, ou descendant. L'idée est ensuite d'utiliser un même ensemble de règles indépendantes de la stratégie, qui en s'appliquant sur tel ou tel automate réaliserait telle ou telle stratégie d'analyse. En pratique, cependant, réaliser un analyseur indépendant de la stratégie est très difficile. Le choix de la stratégie ne repose pas uniquement dans la seule définition du parcours de la structure servant de guide à l'analyse syntaxique. Il n'existe pas encore d'*item* tabulaire capable d'être utilisé par l'ensemble des stratégies d'analyse de LTAG. L'automate SD-2SA (*Strongly Driven 2-Stack Automata*) de [dlC et al.98], qui est complexe, est à notre connaissance le seul modèle capable de mettre en œuvre une stratégie d'analyse de LTAG aussi bien ascendante que descendante. Aussi préférons-nous à cette approche la compilation d'un unique automate sur lequel il sera possible de mener toutes les stratégies voulues en fonction de l'ensemble de règles employées, et sur lequel il sera possible de retrouver les parcours connexes le composant.

Prenons ainsi l'exemple des trois schémes de la table 4.1. Il est clair, par exemple, que le rattachement structurel du sujet peut être factorisé à ces trois arbres durant l'analyse syntaxique. Notre technique consiste à représenter chacune des linéarisations de ces arbres par un automate, suivant la technique introduite au chapitre 1 de cette partie, comme l'illustre la figure 4.3. Ensuite, on combine tous les états initiaux et tous les états finaux de ces automates et on minimalise l'ensemble, résultant en l'automate de la figure 4.4. Il existe de nombreux algorithmes classiques de minimalisation qui permettent d'obtenir un tel automate [Aho et al.72].

4.1.6 Adaptation de l'algorithme d'analyse

Un tel automate minimalisé permet d'obtenir pour les algorithmes par *chart* l'équivalent du partage de préfixe des analyses LR quel que soit le sens d'analyse. Lorsque l'ensemble des automates représentants la linéarisation des arbres élémentaires est minimalisé en un

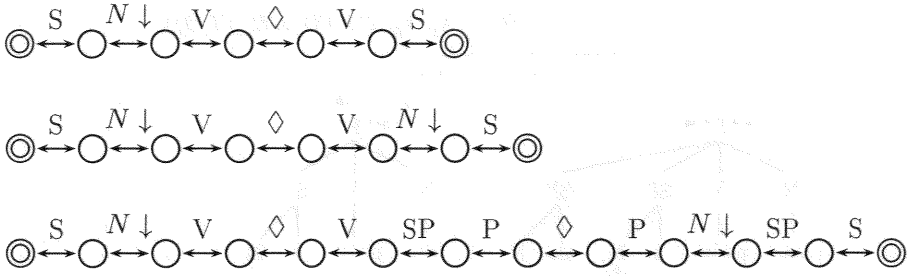


FIG. 4.3 – *FSA correspondant aux schèmes de la table 4.1.*

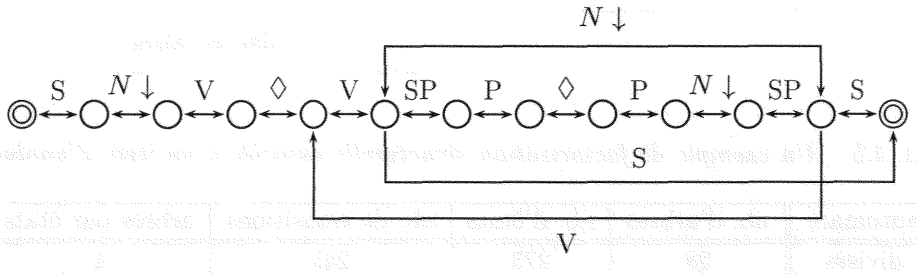


FIG. 4.4 – *FSA correspondant à la compaction des schèmes de la table 4.1.*

seul automate, nous plaçons dans chaque état les listes des identifieurs des arbres élémentaires passant par cet état. Chaque *item* référant à deux états de l'automate général contient également la liste des identifieurs d'arbres élémentaires valides pour les positions d'analyse données par l'*item*. Cette liste d'identifieurs associés à un *item* est toujours un sous-ensemble des identifieurs d'arbres élémentaires passant par les deux états référés par l'*item*. Grâce à cette liste, il est possible de retrouver tous les arbres élémentaires factorisés par un *item* lors du processus d'extraction des dérivations, qui effectue donc une sorte de décompaction du partage structurel de la grammaire LTAG. La difficulté essentielle est que, pour tester l'application d'une règle d'inférence, il est nécessaire de considérer toutes les transitions possibles à partir des états référés par les *items* compte tenu des parcours connexes et de l'automate de partage. L'*item* résultant de l'application d'une règle peut, par la suite, servir pour factoriser d'autres structures ayant atteint les mêmes indices sur la chaîne d'entrée et les deux états référés.

La figure 4.5 donne un exemple d'un tel *item* factorisant une position d'analyse pour un sous-ensemble d'arbres élémentaires[47] d'une grammaire LTAG (sous-ensemble de cardinal deux ici).

4.2 Premières expérimentations

La figure 4.6 donne l'automate minimalisé permettant le partage de 28 arbres élémentaires relatifs à des contextes verbaux intransitifs et transitifs. Le nombre dans les états de l'automate indique combien d'arbres passent par cet état. La table 4.2 donne les résultats

47. Sur cet exemple, l'identifieur $I3\_VD\_3$ réfère à l'arbre initial contenant trois ancres et représentant le contexte verbal ditransitif numéro 3. L'identifieur $I2\_VT\_3$ réfère à l'arbre initial contenant deux ancres et représentant le contexte verbal transitif numéro 3.

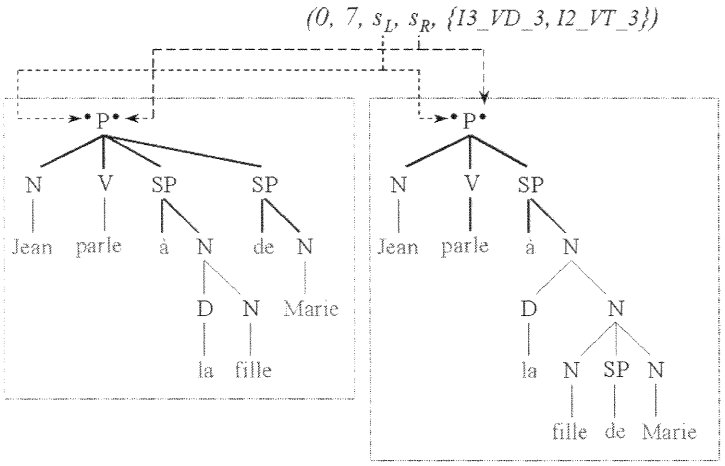


FIG. 4.5 – *Un exemple de factorisation structurelle associée à un item d'analyse.*

automate	nb. d'arbres	nb. d'états	nb. de transitions	arbres par états
divisés	28	273	245	1
partagés	28	13	23	21.84

TAB. 4.2 – *Résultat d'une compaction de grammaire lexicalisée d'arbres.*

de la compaction. Le taux de compaction en nombre d'états est supérieur à un facteur 20 donnant un idée du taux de compaction des opérations pouvant être obtenu. Par exemple si on suppose une adjonction sur le nœud pré-terminal de l'ancre principale verbale, la reconnaissance de l'adjonction pourra être partagée par un nombre d'arbres élémentaires pouvant aller jusqu'à 28, alors qu'elle aurait fait l'objet d'autant d'évaluations séparées sans partage structurel.

Notons que le taux de compaction obtenu par [Evans et al.98] est du même ordre sur une grammaire LDTSG (Lexicalized D-Tree Substitution Grammar) à large couverture de l'anglais que sur cet exemple. Nous n'avons pas encore adapté notre algorithme d'analyse à ce type d'automate minimalisé. L'algorithme ne fonctionne actuellement que sur des automates séparés. Nous ne pouvons pas fournir d'évaluation du gain apporté durant l'analyse. Cependant, des résultats très récents présentés dans [Carroll et al.99b] montrent que le gain en temps durant l'analyse par *chart* est d'un facteur supérieur à 80 pour un algorithme de type CKY adapté à l'analyse d'une grammaire LDTSG compilée sous la forme d'un automate de type [Evans et al.97].

4.3 Automate minimalisé et automate de recherche

Un des avantages de l'automate minimalisé obtenu est qu'il peut être facilement utilisé comme automate de recherche, notamment en vue de rendre déterministe l'analyse. Suivant les arbres et les nœuds les plus fréquemment présents sur un corpus de test, il est possible de pondérer chaque transition de l'automate. Au cours de l'analyse on peut alors considérer en premier les sous-structures les plus fréquentes ou les plus partagées.

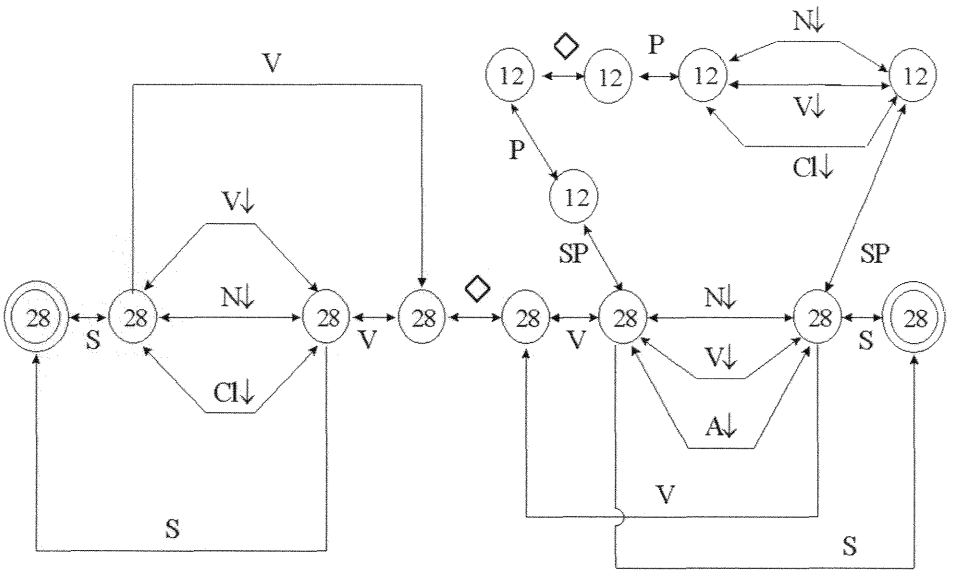


FIG. 4.6 – *FSA représentant 28 arbres élémentaires correspondant à des contextes verbaux intransifs et transitifs.*

Un autre intérêt est que l'automate minimal obtenu donne une vue compacte de la grammaire, sur laquelle il est possible de mener efficacement des recherches. C'est le cas en particulier pour retrouver l'ensemble des arbres élémentaires partageant un motif particulier. De tels motifs s'expriment simplement sous la forme d'une expression régulière. A partir de cette expression régulière, il est possible de construire l'automate équivalent qu'il suffira de composer avec l'automate de la grammaire LTAG par intersection. L'automate résultant donnera sous une forme partagée l'ensemble des arbres élémentaires comportant ce motif. Cette propriété peut être utile en cas d'ellipses pour retrouver les structures antécédentes possibles (*cf. supra* partie 4 paragraphe 2.4.5).

Quatrième partie

De l'analyse de l'écrit à celle de l'oral

Chapitre 1

Limites du formalisme TAG et de l'analyse par connexité

Les formalismes syntaxiques ont pour objectif une description de la compétence linguistique à l'aide d'un nombre réduit d'opérateurs et de principes. Or, la langue orale employée au quotidien présente une variabilité des constructions syntaxiques possibles telle qu'il semble irréaliste de toutes les énumérer. Répétition ou reprise, par exemple, peuvent intervenir à tout moment dans un énoncé. Un formalisme spécifiquement développé pour l'oral se révèlerait sans doute trop lâche et finirait par accepter toutes les constructions possibles. Nous allons montrer dans cette partie qu'en effet, une majorité des erreurs de l'analyse du corpus GOCAD à l'aide d'une grammaire LTAG sont liées à des phénomènes oraux. Mais, compte tenu de la nature d'un formalisme linguistique, peut-on vraiment considérer ces erreurs comme des limites du modèle LTAG qu'il faudrait alors remettre en cause?

1.1 Typologie des phénomènes TAG-agrammaticaux

Une taxonomie des erreurs d'analyse observées sur les transcriptions de dialogues oraux montre que la majorité des erreurs sont liées à des phénomènes oraux spécifiques. La table 1.1 donne les occurrences de ces phénomènes oraux dans le corpus GOCAD, suite à l'analyse présentée dans le chapitre 3 de la partie précédente. Dans les chiffres présentés, plus d'un phénomène peut apparaître dans un même énoncé. Les hésitations, tout d'abord, sont massives : 14,3% des énoncés en comportent. Les répétitions comme (43), présentes dans 3,2% des énoncés, sont également non négligeables. Nous avons regroupé sous la dénomination *reprise* à la fois les reprises introduisant une précision et celles introduisant une correction. Elles sont présentes dans 2,7% des énoncés. 14 des 23 cas constituent des reprises avec interruption du *reparandum* comme pour (44). Un nombre significatif d'ellipses de tête prédicative sont également observées (1,8% des énoncés), conduisant l'analyse syntaxique mise en œuvre à l'échec. Nous les avons assimilé à des phénomènes oraux de façon quelque peu abusive, mais elles nous semblent, dans ce contexte, correspondre à un mode de production caractéristique de l'oral. On retrouve également, de manière peu fréquente, des relâchements grammaticaux concernant essentiellement les accords (45).

(43) *Peut-on changer l'angle de la caméra pour revenir* **à à** *une vue à 90 degrés? (GOCAD)*

(44) *Revenir* **au à** *la vue initiale. (GOCAD)*

(45) *Sur la surface A euh non* **la surface euh B et C**. *(GOCAD)*

Nous avons pu observer 196 occurrences de phénomènes s'inscrivant dans une de ces différentes catégories. 152 énoncés présentent un ou plusieurs de ces phénomènes, soit 17,6% des énoncés du corpus. Les phénomènes oraux se retrouvent dans 82% des énoncés n'aboutissant pas à au moins une analyse complète. Notons que d'autres phénomènes non liés à l'oralité peuvent aussi amener des erreurs parmi ces énoncés problématiques, en plus de phénomènes oraux.

énoncés mal-formés	avec hésitation	avec répétition	avec reprise	ellipse non prévues
occurrences	123	28	23	15

TAB. 1.1 – *Occurrences des phénomènes oraux conduisant à des erreurs d'analyse pour le corpus GOCAD.*

Les autres cas d'erreurs sont liés aux énumérations (46), présentes dans 16 des énoncés et constituent un phénomène non couvert par notre grammaire. D'autres cas d'erreurs proviennent de problèmes d'interruption de l'énoncé (47) (48), de phénomènes de simplification syntaxique de type *petit nègre* (4 cas) (49) et de parataxes (50).

(46) *Hmm peut-on charger les fichiers Pl1 point ts Pl2 point ts Pl3 point ts et Pl4 point ts? (GOCAD)*

(47) *Oui relier les lignes et... (GOCAD)*

(48) *Définir un... (GOCAD)*

(49) *Ajuster surface aux points. (GOCAD)*

(50) *Raffiner le maillage, euh, dans la zone précisée par la souris, effectuer d'abord un zoom un zoom. (GOCAD)*

1.2 Limites descriptives du modèle TAG

Il est connu que certains phénomènes se représentent difficilement à l'aide d'une structure arborescente. C'est le cas des coordinations, que l'on peut envisager cependant de prendre en compte directement au cours de l'analyse syntaxique [Mela92] [Sarkar et al.96].

Les hésitations et les incises pourraient être traitées directement par le formalisme LTAG en tant qu'arbres auxiliaires capables de s'adjoindre à tout endroit. Une telle solution apparaît peu satisfaisante car elle multiplierait la combinatoire d'analyse, chaque possibilité d'adjonction d'une hésitation constituant une ambiguïté supplémentaire. C'est aussi pour cette raison que nous n'avons pas couvert les énumérations par un traitement direct.

Par contre, les répétitions et les reprises semblent échapper à la possibilité d'un traitement direct. La variabilité des constituants répétés et repris, ainsi que l'impossibilité de prédire ces phénomènes laissent penser qu'il faudrait pour toute entrée lexicale considérer un contexte de répétition et de reprise et un contexte d'utilisation « normal ». Les ambiguïtés artificielles que cette solution engendrerait, ainsi que la perte de tout pouvoir prédictif de la grammaire, font que ce choix apparaît comme peu raisonnable.

En tant que formalisme pour l'écrit, une structure LTAG ne permet pas de représenter certains phénomènes. C'est le cas des ellipses de tête prédicative où un modifieur gauche

se retrouve adjacent à un modifieur droit sans possibilité de rattachement. Le premier constituant de l'exemple (51), tiré du corpus Wolff [Wolff99], serait par exemple impossible à reconnaître.

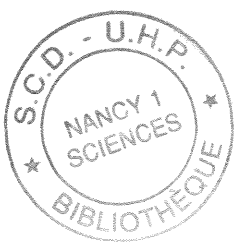
(51) **Les deux restants** *dans l'avant-dernière boîte. (Corpus Wolff)*

Dans la mesure du possible, nous proposons de définir des règles permettant de rattraper les limites formelles de représentation du formalisme, puis d'adapter les TAG pour rendre compte des contraintes de la langue orale sur des structures de l'écrit.

1.3 Limites de l'analyse grammaticale

Les limites de l'analyse grammaticale menée sur la base d'une grammaire LTAG sont celles qu'on pourrait retrouver avec d'autres formalismes syntaxiques. Les résultats partiels, que l'on doit obligatoirement exploiter à l'oral, restent, en particulier, très nombreux. Ce nombre est nuisible à une exploitation ultérieure par composition de segments. Il est alors nécessaire d'employer des heuristiques, par exemple le choix des segments les plus étendus [Abney91], des scores de probabilités et de reclassement [Srinivas et al.95] [Roussel et al.99a] ou d'exploiter des contraintes sémantiques et pragmatiques. Ce dernier point, qui repose sur une interaction plus systématique entre les différents niveaux de traitement, est classique en analyse syntaxique et n'est pas propre à l'oral.

Comme indiqué dans le paragraphe précédent, il semble très difficile de prendre en compte les phénomènes oraux directement dans le processus d'analyse grammaticale. Nous considérons cependant que nous pouvons exploiter ces limites qui sont, en fait, prévisibles étant donné le rôle d'un formalisme syntaxique. Nous avons vu que l'arrêt de l'analyse grammaticale peut signifier une variation paradigmatique souvent liée à l'oral. Nous proposons d'exploiter les limites du formalisme pour l'identification des parties syntaxiquement normalisées des énoncés oraux. Ces parties restent largement majoritaires et il serait à notre sens maladroit de les ignorer. Le cadre d'analyse proposé se fonde donc sur un traitement direct de l'axe syntagmatique des énoncés à l'aide du formalisme LTAG. Les phénomènes d'ordre paradigmatique sont pris en compte indépendamment. Le formalisme a donc un double rôle : en plus d'une analyse syntagmatique incluant les ellipses (chapitre 2 de cette partie), l'arrêt de l'analyse est exploité pour localiser les variations paradigmatiques. Ces dernières font alors l'objet de règles additionnelles spécifiques (chapitre 3). Du point de vue de la prédictivité, la grammaire caractérise un premier niveau de langue normalisée qui reste la plus courante. Sur la base d'indices prosodiques, intonatifs et de durée, nous pensons qu'il est possible d'utiliser les règles additionnelles pour la prédiction et la reconnaissance de phénomènes oraux. Ce dernier point correspond à un relâchement de la prédictivité grammaticale et à l'emploi d'un autre niveau de langue. Ce niveau de langue est considéré, dans ce cadre, comme agrammatical et analysé en tant que tel. Le chapitre 3 de la $5^{ème}$ partie de ce mémoire établit les bases d'une architecture pour un système de dialogue intégrant notre analyseur. Cette architecture a pour but de permettre la mise en œuvre d'une telle stratégie prédictive qui s'inscrit dans la lignée de celles développées au SRI [Price et al.89].



Chapitre 2

Enrichissement des descriptions lexicales

2.1 Introduction

Un des enjeux majeurs de l'analyse syntaxique appliquée au dialogue homme-machine est de pouvoir traiter les énoncés incomplets normalement utilisés par les utilisateurs. En particulier, des études empiriques [Carbonell83] ont montré que, si l'utilisation d'énoncés très fragmentaires était chose courante dans la communication entre humains, elle l'était tout autant en situation de dialogue homme-machine. De plus, bien que les utilisateurs puissent facilement éviter l'emploi de structures syntaxiques complexes, ils peuvent très difficilement se contraindre à l'emploi exclusif d'expressions complètes.

Afin de rester robuste à ce type de phénomène et à d'autres spécifiques de l'oral (reprises, répétitions, etc.), des techniques d'analyse superficielle ou stochastique ont été employées visant à extraire avant tout l'information utile. En effet, dans la mesure où ces phénomènes sont difficiles à prédire, il peut sembler inutile de s'attacher à une grammaticalité jugée illusoire en situation de dialogue oral. Si des techniques d'analyse robuste restent nécessaires pour extraire des énoncés un maximum d'information même si une analyse complète échoue, un effort de modélisation des contraintes orales portant sur les énoncés de l'utilisateur nous semble nécessaire. Une des difficultés de cette modélisation est qu'en se conformant à une théorie grammaticale particulière, on se conforme aux principes de correction et de complétude qui fondent en fait cette théorie. Par exemple il n'est pas prévu pour les LTAG que les arbres auxiliaires puissent être considérés comme constituants complets. En pratique, il ne sera pas possible de substituer un constituant dont la tête nominale est elliptique.

Ce chapitre définit des principes additionnels en vue d'adapter une LTAG conçue initialement pour l'écrit à l'analyse d'énoncés oraux et incomplets. Nous l'avons vu, les phénomènes d'incomplétude ne sont pas de même nature que les autres phénomènes oraux, comme les répétitions, les reprises et les corrections. Ces derniers cas correspondent à un surplus de matériel syntaxique lors de l'analyse. Notre point de vue est que les contraintes portant sur l'incomplétude d'une structure syntaxique peuvent s'exprimer directement dans la grammaire, indépendamment du processus d'analyse. Des expérimentations sur le corpus GOCAD permettent d'évaluer l'intérêt de l'approche proposée.

2.2 Une vue empirique sur les énoncés incomplets

Nous présentons ici le résultat de l'analyse des énoncés utilisateurs du corpus GOCAD. Nous rappelons que nous avons utilisé une LTAG suivant les principes définis dans [Abeille91b], donc une grammaire conçue pour l'analyse de l'écrit et l'algorithme d'analyse par connexité. Nous nous sommes d'autre part fondé sur les critères de l'écrit pour obtenir une analyse complète (la catégorie P est l'unique axiome et la solution ne doit pas présenter de nœuds pieds ou de nœuds de substitution non saturés). La table 2.1 présente également le nombre moyen d'analyses partielles[48] obtenues en fin d'analyse.

Nb moyen de mots par énoncé	% d'analyses complètes	Nb moyen d'analyses/énoncé	Nb moyen de résultats partiels/énoncé
6,4	64,7	1,5	7,1

TAB. 2.1 – *Le corpus GOCAD et son analyse avec une grammaire LTAG pour l'écrit et selon les critères de l'écrit.*

Les résultats présentés table 2.1 montrent que la proportion de phrases complètement analysées est relativement faible. Une taxonomie des échecs d'analyse montre que 50,9% des erreurs viennent uniquement d'énoncés incomplets au sens de l'écrit, le reste pouvant être attribué à d'autres phénomènes (reprises, répétitions, etc.) ou à un phénomène non couvert par la grammaire (énumération par exemple) comme présenté au chapitre précédent. La fréquence des phénomènes d'ellipses nous porte à considérer que leur traitement doit être directement pris en compte durant l'analyse.

2.3 Contraintes minimales sur les énoncés incomplets

Nous étudions ici les ellipses sous l'angle de leur réalisation syntaxique. Bien entendu, la résolution d'une ellipse (« identifier son référent »), comme sa prédiction, dépend du contexte : historique du dialogue, entité saillante, etc. [Carberry89]. Bien que les énoncés de l'utilisateur puissent être très incomplets, des contraintes purement syntaxiques existent cependant dans la langue orale, comme le montrent les échanges suivants :

(52) *Q : Où voulez-vous aller ?*
R : **à** *Sarrebruck.*

(53) *Q : Tu prends ton café avec ou sans sucre ?*
R : **Sans**.

Dans l'exemple (52) la préposition « à » requiert un argument mais « sans » dans (53) peut être utilisé avec ou sans l'introduction d'argument tout en restant une structure syntaxiquement correcte et autonome. Ces exemples montrent d'une part un degré de contraintes minimales assurées à l'oral que nous souhaitons ici pouvoir prendre en compte, et d'autre part que la capture de ces contraintes demande un formalisme fortement lexicalisé puisque le contexte syntaxique minimal varie ici d'une préposition à l'autre.

Nous nous intéressons ici aux ellipses rencontrées dans le cadre de dialogue de commande finalisé et de couple question/réponse. Les énoncés présentent alors un nombre

48. Une analyse partielle correspond ici à l'extension maximale d'un îlot bien reconnu, donc, dans le cadre de l'analyse par *chart* mise en œuvre, à un *item* qui n'est origine d'aucun autre *item*.

important d'ellipses sur la tête prédicative des constituants que l'on peut illustrer avec les exemples suivants :

(54) *système : Le milieu de quoi ?*
utilisateur : **du** *carré.*

(55) *système : Quel objet ?*
utilisateur : **le petit.**

Au vue de ces exemples, il semble donc possible d'exprimer des contraintes purement syntaxiques concernant la verbalisation d'une ellipse. Cependant les cas particuliers d'*emploi en mention* échappent à toute contrainte syntaxique. Dans la mesure où tout mot est susceptible d'être ainsi utilisé, comme le montre l'exemple (56), les contraintes que nous souhaitons représenter ici excluent donc ce style d'emploi[49].

(56) *Q : Vous avez dit prendre, pendre ou fendre ?*
R : prendre.

2.4 LTAG et structures elliptiques

Des grammaires LTAG à large couverture ont été implantées pour l'anglais [Doran et al.94] et le français [Abeille et al.94] mais dans l'optique de l'analyse de textes écrits. Dans la mesure où ces grammaires ne sont pas elliptiques, l'analyse des exemples présentés serait impossible. Cela signifie donc qu'ayant échoué sur le plan syntaxique, leur interprétation sera obtenue uniquement par la sémantique, le risque étant une combinatoire prohibitive. La question de la prédictivité d'une LTAG est importante dans la mesure où cette caractéristique permet de réduire le nombre des hypothèses artificielles et donc la complexité moyenne des analyses.

Couvrir les ellipses augmente le nombre des contextes syntaxiques à prendre en compte au cours de l'analyse. Un des problèmes essentiels est ici de concevoir une grammaire elliptique sans augmenter dramatiquement le nombre d'arbres élémentaires. En particulier, pour le corpus GOCAD, les 64,7% d'énoncés non elliptiques correctement analysés verraient leur complexité moyenne d'analyse sensiblement augmenter. Certains principes définis par [Abeille91b] concernent la relation entre un arbre élémentaire et la sémantique, en particulier un arbre élémentaire ne doit correspondre qu'à un unique prédicat (non nul). En considérant ces principes, on peut envisager deux premières solutions pour analyser des énoncés fragmentaires en évitant autant que possible la multiplication des arbres élémentaires :

- modifier le mécanisme d'analyse en rendant la substitution facultative, les nœuds non saturés étant autant de marques d'ellipses. Cependant les exemples (52) et (53) montrent que le lexique doit indiquer clairement les possibilités structurelles d'ellipses afin de contraindre l'aspect compositionnel ;
- adapter une grammaire non elliptique en considérant toute catégorie comme axiome. Mais ceci aboutira à une multiplication des ambiguïtés artificielles, si on veut rester prédictif on ne peut pas considérer toute structure (par exemple un déterminant) comme complète.

49. Dans les corpus de DHM simulé, on remarque que les emplois en mention n'apparaissent qu'à l'initiative du système, après des questions précises donnant le choix d'un mot parmi une liste fermée. Ce cas particulier peut être évité par l'emploi d'un rang ou couvert par des heuristiques spécifiques lors d'une première étape d'analyse lexicale en fonction de l'état du dialogue. Notons enfin que ce phénomène n'apparaît pas dans le corpus GOCAD.

Les résultats obtenus en se fondant sur ces deux critères sont présentés dans le tableau 2.2. On constate que la proportion d'analyses complètes obtenues passe de 65,9% à 78,3%, 15 cas d'ellipses (sur les 155 initiaux) échappant à la grammaire mais pouvant être récupérés par un mécanisme de rattrapage. Néanmoins, si la couverture est bonne, l'aspect prédictif est ignoré, entraînant une surgénération des analyses. En effet, le nombre moyen d'arbres de dérivation complets par énoncé passe ainsi de 1,5 à 2. De plus, ces choix sous-entendent que toute structure partielle est acceptable à l'oral, ce qui n'est pas forcément le cas, comme le montre l'exemple 52.

% d'analyses complètes	Nb moyen d'analyses/énoncé
78,3	2,0

TAB. 2.2 – *Le corpus GOCAD et son analyse avec une grammaire LTAG pour l'écrit, mais en acceptant les structures incomplètes et ne dérivant pas de l'axiome.*

2.4.1 Limite du formalisme

Le formalisme LTAG emploie deux types d'opérations pour la combinaison des arbres élémentaires, l'adjonction et la substitution, chacune d'elle pouvant être associée à deux rôles, en considérant les choix linguistiques de [Abeille91b] :

- un rôle formel : les substitutions sont obligatoires et représentent donc les incomplétudes d'une structure, les adjonctions sont optionnelles et représentent des rattachements facultatifs ;
- un rôle linguistique : le rattachement d'un modifieur se réalise par adjonction, celui des arguments par substitution.

Si on considère l'analyse de structures incomplètes telles qu'observées en situation de dialogue homme-machine, il est possible d'avoir des arguments non obligatoires, par exemple (53), et des modifieurs apparaissant seuls (55). Ceci fait que le rôle formel devient incompatible avec le rôle linguistique. Nous proposons dans la section suivante un enrichissement du formalisme permettant de préserver le rôle linguistique.

2.4.2 Enrichissement du formalisme

Afin de préserver l'interprétation linguistique des dérivations, nous utilisons toujours un arbre auxiliaire pour représenter les modifieurs et la substitution pour les rattachements prédicats arguments selon les critères présentés par exemple dans [Candito99]. Nous souhaitons de plus représenter que l'occurrence d'un mot implique celle d'une structure syntaxique de façon obligatoire ou optionnelle. Par exemple l'occurrence d'un déterminant implique celle d'un nom, mais la présence d'un nom n'implique pas obligatoirement celle d'un déterminant. Nous nous appuyons en conséquence sur deux marques : $\uparrow$ qui implique un caractère d'obligation, et $\downarrow$ qui marque un caractère optionnel. Nous introduisons alors les annotations suivantes au niveau des nœuds terminaux :

- $\downarrow$: le nœud peut accueillir de manière *optionnelle* une substitution d'un constituant argument, voir exemples figure 2.1 (a) ;
- $\downarrow\uparrow$: substitution *obligatoire* d'un constituant argument (b) ;

- $*$: adjonction *optionnelle* du modifieur, voir (a) et (f) ;
- $*\uparrow$: adjonction *obligatoire* du modifieur, voir (e) ;
- ancre simple : occurrence *obligatoire* d'une ancre dans l'énoncé (cas général) ;
- ancre $\downarrow$: occurrence *optionnelle* d'une ancre dans l'énoncé[50], voir 2.2.

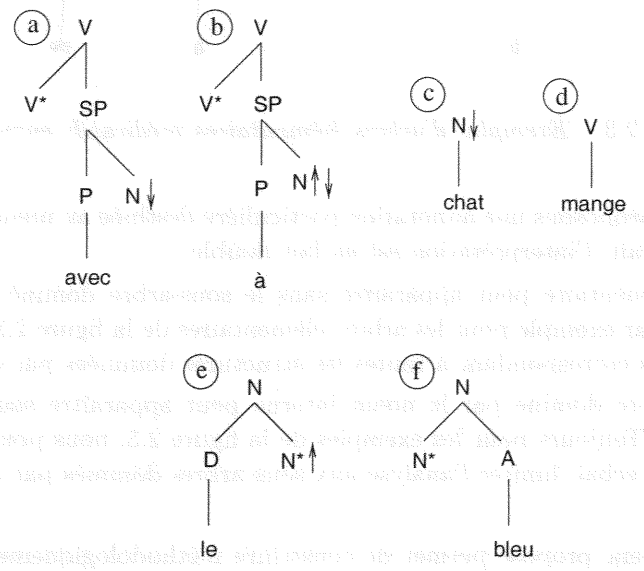


FIG. 2.1 – *Exemples d'arbres élémentaires enrichis.*

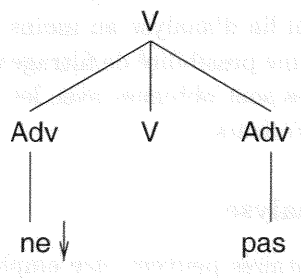


FIG. 2.2 – *Exemple d'effacement dans un arbre élémentaire enrichi.*

Un constituant peut être complet mais toutefois être employé avec une ellipse du prédicat dont il est argument. Afin d'aider des niveaux supérieurs de traitement dans la résolution de ce style d'ellipses nous marquons les nœuds racines de la façon suivante :

- $\downarrow$: le constituant est obligatoirement argument d'un prédicat, voir par exemple 2.1 (c) ;
- pas de marque : le constituant est argument de manière optionnelle, voir (d).

50. Ce cas peut être utile dans un contexte oral pour représenter les effacements, par exemple la chute du discordantiel *ne* dans le cas d'une négation, comme illustré par la figure 2.2.

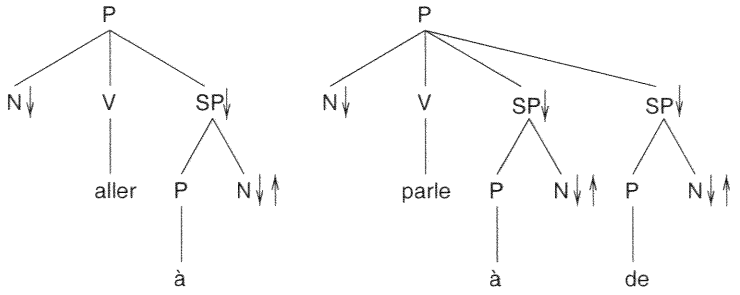


FIG. 2.3 – *Exemples d'arbres élémentaires prédicatifs enrichis.*

Enfin nous proposons une annotation particulière destinée au nœud interne. Lorsqu'une marque ↓ apparaît, l'interprétation est en fait double :

- l'arbre élémentaire peut apparaître sans le sous-arbre dominé par le nœud interne marqué. Par exemple pour les arbres élémentaires de la figure 2.3, nous pouvons avoir des ellipses correspondant à toutes les structures dominées par des nœuds SP ;
- le sous-arbre dominé par le nœud interne peut apparaître seul sans le reste de la structure. Toujours pour les exemples de la figure 2.3, nous pouvons, en cas d'ellipse du noyau verbal, limiter l'analyse aux sous-arbres dominés par les nœuds SP.

L'enrichissement proposé permet de construire méthodologiquement un lexique LTAG représentant les contraintes portant sur les énoncés incomplets, ou d'en adapter un existant. Il suppose que les propriétés de complétude des constituants sont des propriétés lexicales indépendantes du contexte d'analyse. Compte tenu de ces annotations, nous pouvons introduire le principe de complétude d'une analyse d'un énoncé oral de la façon suivante : tout arbre dérivé présentant en fin d'analyse au moins une marque ↑ ne constitue pas une analyse acceptable. Ceci offre une possibilité de filtrage des analyses partielles. La détection et la caractérisation des ellipses sont obtenues avec les nœuds présentant les marques $*$ ou ↓, facilitant les traitements ultérieurs.

2.4.3 Exploitation pour l'analyse

Deux solutions complémentaires peuvent être employées pour exploiter ces notations : soit durant l'analyse, soit en fin d'analyse.

La première solution, la plus naturelle, consiste à une exploitation directe durant l'analyse : la seule modification de l'algorithme d'analyse que nous avons proposé consiste en une adaptation de la définition des parcours connexes. Un parcours devient un chemin d'automate correspondant à la liste des nœuds parcourus successivement jusqu'à atteindre un nœud décoré d'une flèche ↑. Une telle marque signifie en effet qu'il est obligatoire qu'un rattachement se produise pour pouvoir passer à un autre nœud, c'est-à-dire pour atteindre un autre état. Ceci correspond également à une rupture obligatoire de connexité entre deux sous-chemins adjacents d'un automate représentant un arbre élémentaire. Dans ce cadre, nous autorisons également la substitution d'arbres auxiliaires dont la reconnaissance est complète.

Considérons les deux énoncés 57 et 58. Le principe de complétude d'un énoncé autorisera l'analyse de l'énoncé (57), mais bloquera celle de (58). En effet, le déterminant *le*

présentant une marque ↑, comme tous les déterminants, ne formera jamais à lui tout seul un constituant complet. Le parcours connexe droit de l'îlot d'analyse centré sur *le* restera bloqué au nœud pied $N\uparrow$, comme l'indique la figure 2.4. À l'inverse, le constituant *le premier* pourra être accepté comme complet, les deux parcours connexes d'expansion gauche et droit aboutissant au nœud racine, et pourra se substituer en position sujet de l'arbre de *mange*.

(57) *le petit mange un gâteau.*
(58) *le mange un gâteau.*

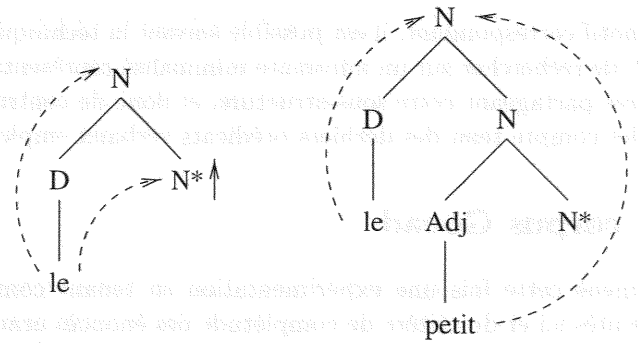


FIG. 2.4 – *Exemples d'expansion au cours d'analyse d'arbres dérivés enrichis*

La seconde solution consiste en une exploitation après analyse. Si une marque ↑ apparaît, on classe l'hypothèse comme agrammaticale.

Pour les deux solutions, il est possible d'appliquer des règles de réparation d'analyse spécifiquement aux résultats considérés comme agrammaticaux. Ces règles de réparation peuvent également exploiter l'enrichissement proposé.

2.4.4 Recherche de structures

De nombreuses ellipses en situation de dialogue portent sur le prédicat verbal, comme pour la dernière intervention de l'utilisateur de l'exemple (59).

(59) *Utilisateur : Tourner cage.*
Compère : Dans quel sens ?
Utilisateur : Vers la droite. (GOCAD)

La résolution de ce type d'ellipse syntaxique suppose qu'on puisse projeter un antécédent de la structure prédicative non réalisée. Cependant la structure prédicat-argument de l'antécédent n'est pas toujours reprise telle quelle dans l'ellipse, ce qui complexifie le processus de résolution. Dans l'exemple, c'est un contexte transitif différent de la première occurrence du verbe *tourner* qui doit être projetée. Il est alors utile de pouvoir faire une recherche des structures élémentaires prédicatives acceptant la sous-structure qui apparaît dans l'énoncé à ellipse. Ici, il faut trouver un verbe acceptant un contexte verbal où l'objet est introduit par une préposition *vers*. Ceci correspond à la sous-structure présentée figure 2.5. On remarque que le nœud racine de cette sous-structure est marqué d'une fléche ↓ indiquant qu'elle peut apparaître, dans un contexte d'ellipse, sans le reste d'une structure prédicative dont elle dépend.

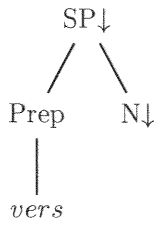


FIG. 2.5 – *Sous-structure LTAG exploitable pour contraindre la recherche d'un antécédent en cas d'ellipse verbale.*

À partir du motif correspondant, il est possible suivant la technique présentée en partie 3 paragraphe 4.2, de rechercher sur un automate minimalisé représentant la grammaire, les arbres élémentaires partageant cette sous-structure, et donc de contraindre une liste d'antécédents possibles compte tenu des derniers prédicats verbaux employés dans le dialogue.

2.5 Retour au corpus Gocad

Nous avons mené cette fois une expérimentation en tenant compte des principes et annotations présentés ici et du critère de complétude des énoncés oraux. Le taux d'analyse complète passe à 79,1% (plus que huit cas d'ellipse sont à prendre en compte par un mécanisme de réparation). De plus, l'analyse présente un gain en termes de réduction de la combinatoire des analyses partielles. On constate que 39,8% des analyses partielles posent un problème et ne forment pas des constituants complets selon les principes introduits. Ceci présente deux intérêts :

- séparer les analyses partielles acceptables des analyses agrammaticales pouvant nécessiter un mécanisme de réparation (répétition, reprises, etc.) ;
- limiter le nombre d'hypothèses à examiner pour contraindre un système de reconnaissance de la parole guidé par une grammaire.

Un point particulièrement important est que ces principes permettent de construire une grammaire elliptique à partir d'une grammaire destinée à des énoncés complets et écrits sans augmenter le nombre d'arbres élémentaires. L'enrichissement consiste uniquement en une série d'annotations sur les arbres existants. Ceci exprime le fait que les possibilités d'ellipses sont des connaissances statiques de la langue. De plus, ceci permet d'être indépendant de l'utilisation de ces connaissances additionnelles. La prise en compte de ces annotations durant l'analyse n'est donc pas obligatoire et peut varier suivant la stratégie adoptée.

Afin de diminuer la combinatoire d'énoncés partiels, la prise en compte au plus tôt des contraintes sémantiques et pragmatiques apparaît nécessaire et peut reposer de manière uniforme sur l'utilisation de grammaires TAG synchrones [Shieber94].

Nous avons présenté une prise en compte des phénomènes d'ellipses et d'effacement dont nous avions souligné l'importance dans la première partie de cette thèse. La solution proposée s'inscrit dans notre cadre général de la prise en compte de l'oral à l'aide du formalisme LTAG. L'expérimentation sur le corpus GOCAD semble très positive : tout d'abord, sur les 155 énoncés présentant au moins un cas d'ellipse, les ellipses de 147 énoncés ont pu être détectées et caractérisées sans provoquer un arrêt de l'analyse. De plus, cette

prise en compte des ellipses n'a pas nécessité d'arbres élémentaires supplémentaires. Les cas d'ellipses posant encore un problème sont des ellipses de tête complexes contenant à la fois des modifieurs normalement antéposés et postposés à la tête syntagmatique. Dans ce cas de figure, la définition même de l'adjonction dans le formalisme LTAG semble poser problème. Plutôt que d'autoriser l'adjonction sur un nœud pied et la substitution d'arbre auxiliaire, nous avons assimilé ces constructions à des réalisations paradigmatiques de syntagme, telle qu'une précision ou une reprise. Nous allons maintenant présenter nos propositions d'analyse de ces variations paradigmatiques au sein de l'analyse par connexité.

Chapitre 3

Règles locales paradigmatiques

Les verbalisations employées étant contraintes à un type de dialogue particulier (dialogue de commande, d'assistance, etc.) et au domaine de l'application, on peut attendre de la part des systèmes de dialogue une certaine robustesse et des traitements opportuns lorsque l'énoncé s'écarte d'une normalisation qui n'est souvent caractéristique que de l'écrit. Lorsqu'une analyse est mise en échec à cause d'un phénomène oral inattendu, comment et sur quels critères rattraper l'analyse? Les propositions présentées ici sont issues d'une réflexion menée avec David Roussel et d'un article commun dont nous reprenons certains éléments [Roussel et al.99b].

3.1 Critères d'appréciation et de correction d'une analyse

3.1.1 Enjeux

Comme l'indique [Antoine et al.99], les variations paradigmatiques dues à l'oral ne semblent obéir à aucune loi statistique. Les meilleurs performances obtenues en reconnaissance de la parole sur des conversations téléphoniques réellement spontanées (corpus Switchboard [Cohen et al.94]) correspondent à des taux d'erreurs lexicales de l'ordre de 50%. Même les méthodes sélectives (voir partie 1, paragraphe 3.4.2) dont l'objectif étaient justement la robustesse à ce type de phénomènes, se révèlent limitées : les répétitions et les corrections représentent une des sources principales d'erreurs de ces systèmes [Minker96]. Pour [Antoine et al.99], il est évident que ce problème connaîtra une importance croissante à mesure que les besoins d'une compréhension détaillée des énoncés se fera sentir.

3.1.2 Analyse stochastique et mécanisme de réparation

Une analyse à l'aide d'une grammaire stochastique peut offrir un traitement des déviations où la probabilité de déviation se combine avec la probabilité d'une analyse. Pour réaliser une analyse stochastique robuste, l'algorithme d'analyse peut être modifié de façon à rechercher une correction probable , donnée par le critère de maximum de vraisemblance. Il faut pouvoir générer la correction la plus probable (notons la y) d'une séquence x. Soit $p(x)$ la probabilité que x appartienne à $L(G)$, le langage généré par une grammaire G, l'algorithme doit calculer $q(y|x)p(x)$, où $q(y|x)$ est la probabilité de déformation de y connaissant x. $q(y|x)p(x)$ est la probabilité de déformation de y sachant $L(G)$. Un tel modèle de déformation stochastique peut être appliqué aux erreurs de substitution [Fung

et al.75] et étendu aux erreurs d'insertion et d'omission. Cependant, les difficultés sont multiples :

- ce mode d'analyse fait l'hypothèse que l'analyse la plus correcte est celle qui s'approche le plus d'une analyse fréquente sur un corpus représentatif. Or, il est difficile d'obtenir une bonne représentation de tous les phénomènes. On ne peut pas généraliser les phénomènes particuliers en les représentant sous forme de classes. En pratique, pour une application donnée, les probabilités ne sont pas toujours stabilisées en accord avec la fréquence des règles dans toutes les analyses correctes syntaxiquement mais avec la fréquence des règles des analyses désambiguïsées également sémantiquement et pragmatiquement. Améliorer les probabilités apprises nécessite d'éliminer les biais sémantiques et liés au contexte pendant la phase d'apprentissage, ce qui est un travail fastidieux car ces analyses ne sont pas vérifiables directement. Leur interprétabilité dépend d'un contexte qui n'est en pratique pas inclus dans le corpus d'apprentissage (en particulier pour les ellipses).
- l'application des probabilités des règles d'une grammaire de réécriture hors contexte est de coût polynomial mais ne permet pas d'obtenir la probabilité d'une analyse, étant donné que ces probabilités ne capturent ni les relations de dépendance entre les règles, ni celles entre les règles et les items lexicaux. Une règle qui peut s'appliquer à différents points de dérivation est privilégiée. Inversement, le nombre de paramètres à manipuler pour une grammaire plus puissante est trop important. [Kahlil97] montre que le problème de désambiguïsation d'une grammaire d'arbres probabiliste est NP difficile si des heuristiques ne sont pas utilisées. Pour éviter l'explosion combinatoire exponentielle dans le modèle DOP [Bod95], un échantillonnage statistique des analyses doit être appliqué ainsi qu'une limitation du nombre de niveaux de profondeur des arbres « élémentaires » (en pratique 4 niveaux maximum).
- plusieurs analyses stochastiques peuvent être maintenues en concurrence mais sans moyen de distinguer, sur la base de probabilités, les analyses obtenues par réparation (substitution, insertion, etc.) ou encore les analyses ambiguës délivrées par une grammaire surgénérative, des analyses grammaticales fréquentes. Un autre problème est de séparer les différentes analyses obtenues par réparation car le score d'une hypothèse à un instant t doit théoriquement prendre en compte non seulement la vraisemblance de la dérivation menée jusque là, mais également la vraisemblance des mots de l'hypothèse à analyser et une estimation du coût de la dérivation. La combinaison de ces trois scores est fragile, étant donné les possibilités de saut ou de réintroduction d'un élément lexical au cours d'une analyse robuste, la vraisemblance de la séquence de mot est biaisée, et l'estimation de l'expansion de l'analyse difficile.

Une conclusion provisoire est qu'une stratégie d'analyse robuste non déterministe requiert un autre modèle de décision. Au lieu de fonder la décision sur des probabilités qui ne reflètent pas une mise en concurrence des différentes analyses, [Roussel et al.98a] proposent d'intégrer les probabilités dans un modèle plus général qui établit une relation de préférence entre plusieurs analyses et remet en cause ce choix en fonction du coût de la prise de décisions. Le choix de l'analyse ayant le moindre coût reste cependant exponentiel. Dans ce cadre, c'est une stratégie d'exploration des îlots qui doit permettre de retrouver une combinaison pertinente efficacement. Pour réaliser la sélection des règles et calculer en priorité la combinaison la plus pertinente, [Lang89] suggère de pondérer les règles. La pondération donne à l'analyseur le moyen de sélectionner l'enchaînement de règles le plus proche des verbalisations prévues ; la grammaire devient un objet qui juge le degré de

normalisation des énoncés. Ceci change considérablement le travail de description d'un langage et pose à nouveau les questions suivantes : cette normalité peut-elle être définie indépendamment du contexte de dialogue ? Peut-on motiver les pondérations en fonction de considérations ergonomiques et linguistiques, complétées par des observations sur un corpus obtenues suivant une méthodologie linguistique à définir ?

3.1.3 Déterminisme et agrammaticalité

Différents niveaux d'ambiguïtés justifient un non-déterminisme de l'analyse des énoncés déviations :

- les ambiguïtés des N-meilleures solutions en sortie du système de reconnaissance de la parole ;
- les ambiguïtés des descriptions syntaxiques, même pour les sous-langages applicatifs (en particulier dialogue de commandes multimodaux) ;
- les ambiguïtés des phénomènes oraux : par exemple entre les constructions syntaxiques des précisions, des corrections et des énumérations, seuls des critères sémantiques et contextuels peuvent trancher l'interprétation.

Du fait de parcours d'espaces de recherche très vastes, une stratégie qui réaliserait en parallèle plusieurs analyses correspondant à plusieurs hypothèses d'un système de reconnaissance est une solution très coûteuse. Une méthode d'analyse comme GLR*, même si elle est reconnue parmi les méthodes d'analyse les plus efficaces ([Lavie96], [Rose et al.97] pour une extension), reste très coûteuse. L'algorithme doit calculer :

- soit la distance minimale en termes de corrections nécessaires entre chaque hypothèse de reconnaissance et l'ensemble des chaînes que définit la grammaire ;
- soit la distance minimale entre une dérivation et toutes celles qui peuvent correspondre à une séquence proche, ce qui suppose de calculer un grand nombre de dérivations.

D'un point de vue théorique aussi, les travaux de [Lavie96] comme ceux de [Lang88] permettent de détecter la proximité d'une analyse robuste avec différentes constructions mais pas de distinguer un énoncé déviant d'un énoncé particulier produit en situation de communication homme-machine. La méthode proposée par [Lang88] permet à partir d'un énoncé incomplet et d'une grammaire G de retrouver les chaînes complètes de $L(G)$ le recouvrant. En pratique cependant, le problème est plus de faire un tri parmi une masse très importante d'hypothèses à la sortie du système de RAP, que d'identifier des éléments manquants. L'algorithme de [Lavie96] est prévu initialement pour trouver une analyse quel que soit le cas, alors qu'on préférerait en pratique rejeter certaines analyses en fonction du type d'erreurs détectées et demander à l'usager de répéter ou de reformuler. Il ne s'agit pas de continuer l'analyse dans tous les cas, surtout lorsque l'incomplétude de la grammaire risque de masquer la reconnaissance d'un énoncé plausible. Il est nécessaire d'admettre que la grammaire n'est pas complète dans toutes les situations de dialogue. Les actes de dialogue assertifs du système sont par exemple exposés à des verbalisations très ouvertes. Autrement dit, une tentative de recouvrement des déviations vers l'énoncé grammatical le plus proche n'est pas toujours pertinente. Admettant cela, il reste néanmoins à examiner quelle stratégie d'analyse robuste développer en référence à une grammaire. Conjointement aux travaux consacrés à la détection d'indices acoustiques, prosodiques et lexicaux et à un traitement de bas niveau des déviations, plusieurs travaux convergent vers une première étape de recherche d'îlots d'analyse syntaxique. Une seconde étape réside ensuite dans l'application d'heuristiques et de principes d'analyse.

3.1.4 Analyses non-syntaxiques

Il est clair que certains phénomènes comme les hésitations et les répétitions ne nécessitent pas une analyse syntaxique pour être détectés. Au contraire, une telle analyse appliquée directement sur ces phénomènes aboutit à des erreurs ou des ambiguïtés supplémentaires. Des processus plus proches de la reconnaissance des formes sur le signal ou de motifs lexicaux simples permettent d'identifier et de corriger ces phénomènes d'une façon relativement fiable. On peut distinguer trois types d'approches correspondantes, de complexité dans les moyens mis en jeu et d'efficacité croissante :

- *pré-reconnaissance* : [Oshaughnessy92], pour les hésitations, et [Nakatani et al.94] ont étudié le signal de parole en vue d'identifier des indices de phénomènes oraux ;
- *pré-analyse syntaxique* et *post-reconnaissance* : suivant cette approche [Heeman et al.94] [Heeman et al.96], par exemple, emploie des techniques de *pattern-matching* de mots, sur la base d'un étiqueteur morpho-syntaxique, mais sans analyse syntaxique. Cette technique tente aussi de s'appliquer à certains types de reprise.

Une variante de cette approche est celle de *post-analyse syntaxique* : dans [Bear et al.92] un *pattern-matching* très lâche donne l'ensemble des points où une distortion de l'oral est susceptible d'apparaître. Ensuite, des techniques d'évaluation d'analyse sont utilisées pour discriminer les phénomènes effectivement à rattraper de ceux acceptables en l'état.

Ces différentes approches se sont concentrées sur les aspects prosodiques et lexicaux des phénomènes oraux. Il est clair que ces caractéristiques doivent être exploitées pour un traitement et une reconnaissance de ces phénomènes. Cependant elles négligent les aspects syntaxiques : par exemple la similarité syntaxique entre le *reparandum* et le constituant de réparation n'est pas vraiment exploitée. L'utilisation d'un étiqueteur morpho-syntaxique dans [Bear et al.92] s'avère insuffisant. En effet, les informations syntaxiques qu'il fournit sont limitées aux catégories, alors que les structures de constituants et les dépendances seraient plus contraignantes pour la détection de tels phénomènes.

Aussi, nous allons maintenant nous intéresser aux approches syntaxiques de la prise en compte des phénomènes déviants.

3.1.5 Des règles syntaxiques pour les phénomènes déviants

Carbonell & Hayes

L'approche présentée dans [Carbonell et al.83] suggère trois méthodes pour détecter les parties « parasites » d'un énoncé et les reprises :

1. lorsqu'une séquence où deux constituants de même type syntaxique et sémantique sont détectés alors qu'un seul est permis, alors ignorer le premier ;
2. reconnaitre explicitement les expressions de correction (par exemple *Je veux dire ...*) et si le constituant directement à droite est de même type syntaxique et sémantique que celui immédiatement à gauche de l'expression de correction, alors ignorer le constituant à gauche et l'expression de correction. ;
3. en réalisant ces corrections, selectionner le constituant minimal sur la gauche, par exemple dans : *Add a high speed tape drive, that's disk drive, to the order, disk drive* doit se substituer seulement à *tape drive* et à non l'ensemble *high speed tape drive.*

Les expressions correctives sont choisies parmi un petit ensemble. Nous formulons deux réserves à ces principes. Tout d'abord, ces derniers sont choisis arbitrairement plutôt qu'en faisant appel à un règle générale. De plus la détection de ces expressions peut amener à

des erreurs d'analyse. Cette approche ne tient pas compte des constituants interrompus, de plus les termes de *type syntaxique* et *type sémantique* sont vagues.

Hindle

Dans [Hindle83], Donald Hindle présente un ensemble de règles en vue de résoudre le problème de l'auto-réparation dans le langage parlé pour son analyseur déterministe *Fidditch*. Il fait l'hypothèse que le point d'interruption est signalé par un marqueur (le signal d'édition) et propose les règles suivantes :

1. *Règle de copie de surface* : une règle lexicale détecte que deux chaînes de mots des deux côtés du marqueur d'interruption sont identiques.
2. *Règle de copie de catégorie* : étant donné une séquence de catégorie du type *[X - signal d'édition - X]*, où X est un constituant, effacer le premier X et le signal d'édition.
3. *Règle de copie de pile* : étant donné une séquence de catégorie du type *[X[ab] - signal d'édition - X]*, où X[int] est un constituant incomplet ([int] signifie une interruption du constituant), effacer le premier X interrompu et le signal d'édition.
4. *Nouveau départ* : la reprise de l'énonciation d'une phrase est un cas spécial, que Hindle justifie par la présence d'une marque corrective du type *vous savez*.
5. *Les phénomènes oraux dégénérés* qui correspondent à des cas où le signal d'édition sont juste ignorés.

Le problème est que ces principes sont destinés à être mis en œuvre dans un analyseur déterministe ce qui suppose d'ordonner ces règles, qui sont en pratique très ambiguës. À un phénomène d'auto-réparation donné par exemple peut correspondre de nombreuses applications différentes des règles.

Levelt

Willem Levelt dans [Levelt89] présente une théorie décrivant les auto-réparations comme une auto-surveillance de sa propre parole. Cette surveillance pouvant avoir lieu à différents niveaux, lexical, morphologique, syntaxique et sémantique, nous assistons effectivement à différents types d'auto-réparation. Pour lui, ces phénomènes doivent être considérés directement comme un processus régulier de l'analyse syntaxique, contrairement à l'approche traditionnelle. Syntaxiquement parlant, il considère les réparations comme un type particulier de coordination qui doivent être prises en compte suivant le même type de règle.

Levelt formule ses principes à l'aide de la règle suivante : la règle de bonne formation des auto-réparations. Celle-ci peut s'énoncer ainsi : un énoncé original plus reparandum *[O R]* est bien formé s'il existe une chaîne C tel que *[O C 'ou' R]* est bien formé.

Pour un exemple de ce principe de bonne formation, considérons l'exemple de l'énoncé (60).

(60) *juste à coté de... juste en dessous de toi.*

Si on pose :

O = 'juste à coté de'
C = 'toi'
R = 'juste en dessous de toi'

On obtiendra la chaîne suivante complète :

[O C 'ou' R] = ' juste à coté de toi ou juste en dessous de toi'

Cette chaîne étant syntaxiquement complète, l'énoncé (60) est acceptable. Ce principe prend en compte des reprises avec constituants interrompues, il présente de ce fait un intérêt particulier. Il peut s'exprimer encore autrement, une fois la syntaxe de la conjonction de coordination *ou* établie dans un formalisme, cette règle devient équivalent à *[O C]* et *[R]* sont deux constituants de même catégorie.

3.1.6 Exploitation de la structure arborescente

Dans notre cadre d'analyse, lorsqu'une analyse complète de la phrase échoue, l'équivalent d'un ensemble d'arbres dérivés et de dérivations partiels est délivré. Les résultats partiels que nous considérons sont les expansions maximales des îlots.

À la différence des grammaires hors contextes, les résultats partiels manipulés conservent à tout moment une forme arborescente. La formalisation présentée dans le chapitre décrivant l'algorithme d'analyse par connexité (chapitre 1 de la partie 3) va nous permettre d'exploiter cette structure arborescente de manière plus contrainte par rapport à un algorithme tabulaire classique où un item est simplement lié à un nœud. Les mécanismes de réparation seront simplement une extension des règles normales d'analyse par connexité.

La représentation d'îlots et des parcours connexes gauche et droit (contexte syntaxique local de l'îlot) que nous avons proposé est plus expressive qu'un *item* classique de type (N,i,j,k,l). En effet, dans les *items* manipulés, les parcours connexes gauche et droit donnent les frontières gauche et droite de l'îlot représenté. Hors, comme présenté précédemment, les règles syntaxiques associées classiquement aux phénomènes déviants correspondent à des contraintes de frontières de constituants. Ces contraintes pourront chez nous s'exprimer directement à partir des *items* d'analyse en tant que contraintes sur les parcours connexes. De la même façon que pour les règles d'analyse grammaticales par connexité, le contexte syntaxique local peut déclencher des règles additionnelles s'appliquant de façon monotone sur le *chart*, mais de nature cette fois paradigmatique. Nous allons maintenant présenter ces règles aditionnelles.

Dans la suite, nous notons $\Rightarrow^*$ la fermeture transitive de la relation d'inférence entre deux *items*. Si i_1 et i_2 sont deux *items* tels que $i_1 \Rightarrow^* i_2$, alors i_2 peut être obtenu à partir de i_1 après lui avoir appliqué un ensemble de règles d'inférence. Le *chart* est noté $\mathcal{C}$.

3.2 Système de règles d'inférence pour les variations paradigmatiques

3.2.1 Traitement des hésitations

Les phénomènes d'hésitation ne nécessitent pas une prise en compte au niveau syntaxique. Cependant, de façon uniforme aux variations paradigmatiques, nous proposons une opération de recouvrement des hésitations, voir table 3.1. Il est utile de conserver la marque d'une hésitation en vue de facilité l'identification et le traitement d'un phénomène plus étendu par d'autres règles de réparation. Cette règle se contente d'absorber la marque d'hésitation identifiée dans le lexique ou par prétraitement à l'aide d'une catégorie préterminale H.

(a) Règle pour les hesitations (règle d'absorption) :

$$\frac{(i,j,\sigma_L,\sigma_R) \qquad (j,k,\sigma'_L,\sigma'_R)}{(i,k,\sigma_L,\sigma_R)} \quad (head(\Gamma'_L) = tail(\Gamma'_R) = H)$$

TAB. 3.1 – *Règle syntaxique de recouvrement des hésitations.*

3.2.2 Précision, correction et énumération

Précision, correction et énumération libre peuvent présenter des configurations syntaxiquement identiques. Seule la sémantique, aidée du contexte, pourra qualifier précisément ces variations paradigmatiques. Nous proposons donc de traiter ces cas de figures à l'aide de deux règles captant cette ambiguïté. La première règle est présentée en 3.2 et illustrée en 3.1 avec un exemple issu du corpus GOCAD. Elle s'applique lorsque deux constituants correctement reconnus et dominés par un nœud de même catégorie se trouvent adjacents et dans l'incapacité de se rattacher à d'autres arbres. La règle (b) produit un *item* recouvrant les deux îlots. Ce nouvel *item* permettra la poursuite de l'analyse, tout en gardant une trace de la réparation effectuée dans l'historique des dérivations.

(b) Règle pour les précisions/correction/énumération :

$$\frac{(i,j,\sigma_L,\sigma_R) \qquad (j,k,\sigma'_L,\sigma'_R)}{(i,k,\sigma_L,\sigma'_R)} \quad \exists X, X \in \Gamma_L \wedge X \in \Gamma_R \quad \wedge \quad X \in \Gamma_L{}' \wedge X \in \Gamma_R{}'$$

TAB. 3.2 – *Règles pour les précisions/corrections/énumérations dans une configuration ambiguë.*

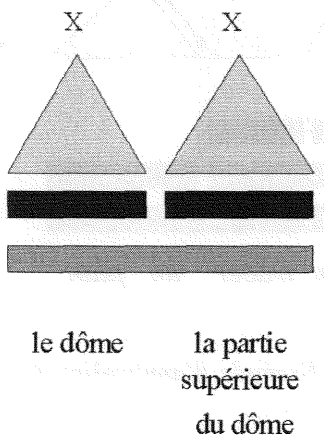


FIG. 3.1 – *Exemple d'application de la règle (b).*

La règle (b) s'applique dans le cas de figure où les deux constituants consécutifs ne se sont pas combinés entre eux, ni avec d'autres arbres. Le plus souvent un des deux constituants s'est déjà rattaché à un autre arbre lorsque la variation paradigmatique est détectée[51]. Pour ce cas de figure, nous employons la règle (c) (table 3.3). Cette règle

51. Par analogie avec le fonctionnement humain, on pourrait dire qu'on construit la phrase à la volée de gauche à droite. Il est donc normal de rattacher un réparandum complet avant la formulation d'une correction remettant en cause le rattachement précédent.

va remettre en cause un rattachement réalisé dans le segment gauche, sur la base de la frontière gauche du segment droit. Si on parvient à retrouver un constituant de même catégorie et adjacent au segment droit, alors il est possible de rassembler l'ensemble en tant que réalisation d'un même constituant. Un exemple d'application de la règle (c) sur un énoncé du corpus GOCAD est illustré par la figure 3.2. Cet énoncé illustre une ambiguïté introduite par l'auto-correction entre une précision et une correction.

La définition de l'auto-réparation donnée dans [Cori et al.97] stipule que la frontière droite de l'arbre dérivé au point de rattachement du *reparandum* doit correspondre syntaxiquement avec la partie gauche de l'élément de reprise (structure d'îlot adjacent). La règle (c) exprime cette condition dans le système d'inférence introduit.

(c) Règle pour les précisions/correction/énumération après rattachement du reparandum :

$$\frac{(i,j,\sigma_L,\sigma_R) \qquad (j,k,\sigma'_L,\sigma'_R)}{(i,k,\sigma_L,\sigma'_R)} \quad \begin{array}{l}(\exists i = (v,j,\sigma_L{''},\sigma_R{''}) \in \mathcal{C},\rangle \Rightarrow^* (),|,\sigma_{\mathcal{L}},\sigma_{\mathcal{R}}) \quad \wedge \\ (head(\Gamma_L{''}) = X^{\triangle} \quad \wedge \quad head(\Gamma'_L) = X^{\triangle}) \vee \\ (head(\Gamma_L{''}) = X* \quad \wedge \quad head(\Gamma'_L) = X*))\end{array}$$

TAB. 3.3 – *Règles pour les précisions/corrections/énumérations dans une configuration ambiguë et avec rattachement du premier segment.*

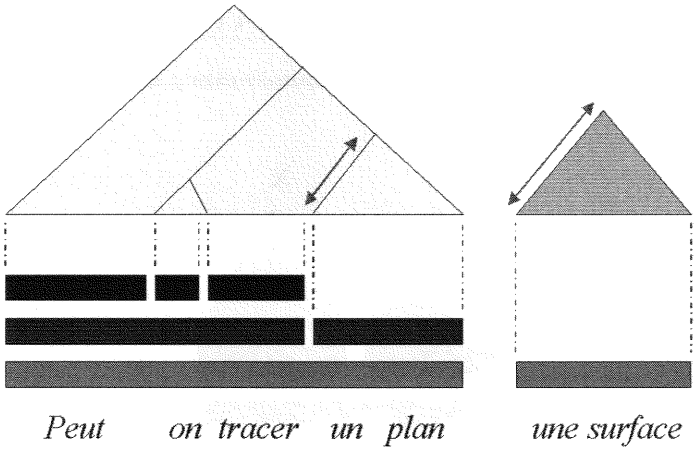


FIG. 3.2 – *Exemple d'application de la règle (c).*

3.2.3 Réparation avec reparandum interrompu

Les règles présentées jusqu'à maintenant supposaient que le *reparandum* était complet. De nombreux exemples montrent que ce n'est pas toujours le cas, par exemple, dans le corpus GOCAD, l'énoncé (61). La prise en compte de ces phénomènes est réalisée à l'aide d'une règle relativement proche du principe énoncé par Levelt (table 3.4).

(61) *Fais une rotation de 30 degrés* **vers** *euh* **du** *bas vers le haut de l'écran. (GOCAD)*

Lorsque ces différentes formes de reprise sont accompagnées d'ellipses ou de marques lexicales venant souligner leur fonction (marques d'acquiescement, d'hésitation, etc.), les

(d) Règle d'autoréparation avec réparandum incomplet :

$$\frac{(i,j,\sigma_L,\sigma_R) \quad (j,k,\sigma'_L,\sigma'_R) \quad (k,l,\sigma_L{}'',\sigma_R{}'')}{(i,l,\sigma_L,\sigma_R{}'')}$$

$$\begin{array}{l}(((tail(\Gamma'_R) = Y* \ \vee \ tail(\Gamma'_R) = Y\downarrow) \ \wedge \\ head(\Gamma'_L) = Y^{\triangle}) \ \wedge \\ (\exists X \in \Gamma_R \ \wedge \\ \quad head(\Gamma_L{}'') = X* \ \wedge \ tail(\Gamma_R{}'') = X^{\triangle}) \ \vee \\ \quad (tail(\Gamma_R) = X\downarrow \ \wedge \ head(\Gamma_L{}'') = X^{\triangle} \ \wedge \\ \quad tail(\Gamma_R{}'') = X^{\triangle}))\end{array}$$

TAB. 3.4 – *Règle de réparation avec réparandum incomplet.*

reprises considérées échappent aux techniques de détection par correspondance de motifs [Bear et al.92], plus appropriées à la détection de répétitions ou d'ajouts. La prise en compte du contexte syntaxique que nous proposons, ainsi que la monotonie d'application des règles, permettent de palier à ces difficultés. L'occurence de plusieurs phénomènes oraux dans une partie d'énoncé proche ou dans un même constituant pourra être traitée par l'application successive des règles correspondantes.

3.2.4 Ellipses vues comme réparation

Plutôt qu'une prise en compte directe durant l'analyse, comme celle proposée au chapitre précédent en 2.4.3, il est possible de traiter les ellipses de têtes syntagmatiques par des règles additionnelles de recouvrement. L'intérêt par rapport à une prise en compte directe est un meilleur contrôle par rapport à un processus systématique. Ces règles peuvent exploiter également l'enrichissement proposé dans le chapitre précédent. L'application des règles (e) et (f) de la table 3.5 correspond syntaxiquement à une transcatégorisation : l'application d'une de ces règles permet à un modifieur d'adopter le comportement syntaxique d'une tête syntagmatique.

Contrairement à (e) et (f), une règle additionnelle nous semble obligatoire pour le recouvrement d'ellipses où un modifieur gauche et un modifieur droit du même constituant se retrouvent adjacents. Nous avons appelé *ellipse complexe* ce cas de figure. Le recouvrement de ce type de constituant est réalisé par la règle (g) et illustré figure 3.3.

(e) Règle de recouvrement des ellipses de tête sur la gauche :

$$\frac{(i,j,\sigma_L,\sigma_R)}{(i,j,previous(\Gamma_L),\sigma_R)} \quad (head(\Gamma_L) = X\downarrow)$$

(f) Règle de recouvrement des ellipses de tête sur la droite :

$$\frac{(i,j,\sigma_L,\sigma_R)}{(i,j,\sigma_L,next(\Gamma'_R))} \quad (tail(\Gamma_R) = X\downarrow)$$

(g) Règle de recouvrement des ellipses complexes de tête :

$$\frac{(i,j,\sigma_L,\sigma_R) \qquad (j,k,\sigma'_L,\sigma'_R)}{(i,k,\sigma_L,\sigma'_R)}$$

$$\begin{array}{l}(tail(\Gamma_R) = X* \ \wedge \ head(\Gamma_L) = X^{\triangle} \\ ((head(\Gamma'_L) = X\downarrow \ \vee \ head(\Gamma'_L) = X*) \ \wedge \\ tail(\Gamma'_R) = X^{\triangle}))\end{array}$$

TAB. 3.5 – *Règles additionnelles de recouvrement des ellipses de tête.*

L'ensemble de ces règles additionnelles présente une complexité au pire des cas en

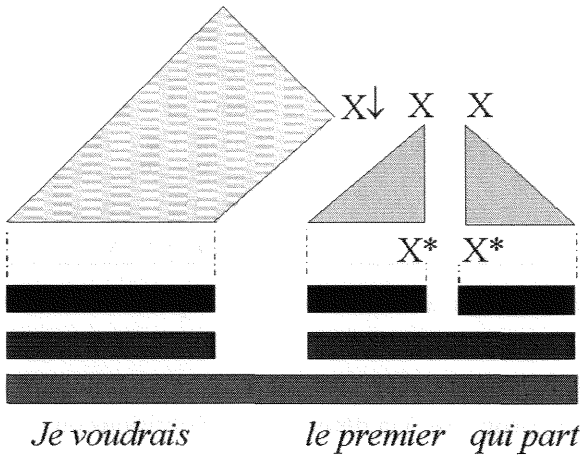


FIG. 3.3 – *Exemple d'application de la règle (g).*

$\mathcal{O}(n^4)$, la complexité maximale provenant de la règle (c).

3.3 Premiers résultats

Les règles ont été intégrées à l'analyseur par connexité existant. Nous employons une stratégie en deux passes : la première est celle de l'analyse syntagmatique normale à l'aide de LTAG dans une stratégie ascendante par connexité, la seconde à l'application des règles paradigmatiques que nous venons de présenter. Cette stratégie sépare l'analyse grammaticale (résultant des substitutions et adjonctions d'arbres) des analyses de phrases potentiellement acceptables mais agrammaticales. Cette stratégie permet de garder un temps d'analyse inchangé pour les énoncés grammaticaux puisque le second ensemble de règles ne s'employera que lorsqu'il n'est plus possible d'étendre les îlots existants.

L'application des règles est monotone sur le *chart*, c'est-à-dire qu'aucune des règles n'est destructrice. Après avoir appliqué une des règles paradigmatiques, on tente l'expansion normale des îlots obtenus jusqu'à parvenir à des résultats complets. Si tel n'est pas le cas, on tente à nouveau l'application d'une des règles du second ensemble et ainsi de suite. Les règles paradigmatiques s'appliquent aux *items* adjacents non rattachables et prioritairement aux segments les plus longs.

Nous présentons dans la table 3.6, les résultats de réparation obtenus sur le corpus GOCAD.

énoncés déviants	avec hésitations	avec répétition	avec auto-réparation	ellipses non-prévues
% correctement rattrapé	79,6	78,5	63,6	46,7

TAB. 3.6 – *Réparations d'analyse pour le corpus GOCAD après application des règles paradigmatiques.*

Le tableau 3.7 donne les résultats globaux d'analyses complètes obtenus sur le corpus

GOCAD. Si on s'en tient aux critères de l'écrit et qu'on tente de n'avoir que des structures complètes dérivées de l'axiome, les résultats sont très insuffisants. En acceptant les structures incomplètes qui ne dérivent pas forcément de l'axiome, on parvient à un taux proche de 78% d'analyses complètes. L'application des règles de rattrapage permet de faire progresser l'analyse et amène à un taux intéressant de résultats complets. Notons que pour les énoncés qui continuent à échapper à des analyses complètes, un traitement sémantique robuste est encore possible à mener et permettrait de retrouver correctement la forme logique de l'énoncé dans un certain nombre de cas.

Analyse	critère de l'écrit	grammaire elliptique	après réparations
% d'analyses complètes	64,7	78,3	93,2

TAB. 3.7 – *Le corpus Gocad et les taux d'analyses complètes avec une grammaire LTAG et des critères pour l'écrit, une grammaire elliptique et enfin après l'application des règles paradigmatiques.*

Le temps moyen d'analyse pour les énoncés du corpus GOCAD sur lesquels s'appliquaient au moins une règle paradigmatique est de l'ordre de 150 ms (dans les mêmes conditions d'expérimentation qu'au chapitre 3 de la partie 3). L'explosion combinatoire en termes d'ambiguïtés est évitée grâce à :

- la stratégie en deux passes ;
- l'application des règles additionnelles uniquement sur des *items* adjacents en cas de blocage de l'analyse ;
- la reprise de l'expansion normale (première passe) après l'application d'une règle paradigmatique ;
- la sélection des *items* les plus étendus.

Discussion

Pour limiter la surgénération durant la deuxième passe, des critères de bonnes formations sémantiques et une méthode fondée sur des scores peuvent être appliqués. Nous considérons qu'à chaque règle paradigmatique de réparation syntaxique correspond une contrepartie sémantique permettant de désambiguïser les phénomènes présentant une même structure syntaxique de surface et d'éliminer si nécessaire les consituants repris. Comme expliqué dans [Lopez et al.98], cette correspondance peut s'appuyer sur des LTAG synchrones.

Lorsqu'aucune réparation n'est parvenue à faire aboutir l'analyse à un recouvrement complet de l'énoncé, il est possible de mener une analyse sémantique robuste sur la base des arbres partiels de dérivation disponibles. Les traitements robustes de ce niveau reposeraient alors sur ce que [BB90] désigne par *macro-syntaxe*.

Nous pensons enfin que l'amélioration de notre proposition passe maintenant par des tests sur un volume plus important de données déjà annotées. Ceci permettrait notamment d'obtenir des scores nécessaires à l'évaluation de la précision et de la correction de nos rattrapages.

Cinquième partie

Le système EGAL : un atelier de conception et de tests de LTAG pour des sous-langages d'application

Chapitre 1

Méthodologie de conception d'un sous-langage

1.1 Motivations

Nous avons pu observer ces dernières années le développement de grammaires lexicalisées à large couverture pour l'écrit. Nous nous sommes intéressé durant les chapitres précédents à l'interprétation de la langue orale spontanée à partir de ce style de grammaires et de techniques additionnelles dédiées aux phénomènes oraux. Aux niveaux lexical et syntaxique, l'application pratique de ces méthodes passe, selon nous, par la possibilité de spécialiser un lexique et une grammaire motivés linguistiquement et destinés à du texte tout venant pour un type de dialogue et un domaine particulier.

Nous nous intéressons dans cette partie à ce problème dont la difficulté justifie parfois l'abandon de modèles linguistiques conçus manuellement pour des analyses plus superficielles et des modèles construits selon des techniques probabilistes. Il faut cependant noter qu'une grammaire écrite manuellement suivant des principes linguistiques permet une compréhension plus détaillée et précise des phénomènes mis en jeu. Les systèmes probabilistes, de plus, nécessitent des corpus d'entraînement très volumineux dont la conception peut en fait relever d'un effort tout aussi important que l'écriture manuelle d'une grammaire.

Nous présentons une méthodologie et un système nommé EGAL (Extraction de Grammaire d'Arbres Lexicalisés), capable d'extraire de façon semi-automatique un sous-langage applicatif à partir d'une grammaire générale et de corpus de type Magicien d'Oz. Une fois la sous-grammaire d'application obtenue, un module d'analyse permet de la mettre à l'épreuve sur un corpus de tests suivant plusieurs algorithmes présentés précédemment et qui s'intégrent à ce système. Les dérivations partielles et complètes peuvent alors être visualisées et comparées considérant différents critères. Ces tests permettent en outre d'obtenir des informations quant à la représentativité du corpus initial utilisé pour décrire le sous-langage. Enfin, grammaire et analyseur sont destinés à être intégrés dans des systèmes de dialogue opérationnels.

Notre objectif est de concevoir des interfaces vocales Homme-Machine génériques et portables permettant la communication en langue spontanée. Pour illustrer la méthodologie et le système proposés, nous avons choisi le corpus GOCAD, dont nous avons extrait un lexique et une grammaire d'application.

1.2 Développement d'une grammaire lexicalisée

La lexicalisation d'un formalisme syntaxique consiste à associer à chaque entrée lexicale une modélisation des contextes syntaxiques dans lesquels elle peut être utilisée. Lexique et grammaire se confondent alors en un lexique syntaxique. La lexicalisation offre la possibilité d'associer à chaque entrée lexicale un degré de finesse des descriptions particulièrement souple, évitant les effets de bord de grammaires fondées sur une régularité de la langue.

Nous avons vu que la lexicalisation présentait cependant certaines contreparties au niveau performance d'analyse (chapitre 4 de la partie 3). Une autre contrepartie est la tâche de conception d'une grammaire. Encore en cours d'amélioration, la grammaire anglaise du système XTAG [Doran et al.94] a d'ores et déjà demandé plus de sept ans de développement, celle du français [Abeille et al.94] plus de cinq ans. Une grammaire à large couverture peut contenir plus d'un millier de schèmes [Candito99], et nécessite la conception d'une base de données syntaxiques décrivant pour chaque lemme les arbres ou les familles d'arbres correspondants. En considérant une application donnée, utiliser une grammaire complète de la langue aboutirait à un nombre prohibitif d'hypothèses. D'autre part notre but est d'éviter d'avoir à concevoir entièrement une nouvelle grammaire spécifique pour chaque application.

Différents travaux sur l'utilisation de LTAG dans des systèmes de dialogue ont été récemment menés [Roussel et al.98b][Halber98b][Lopez98a] mais l'adaptation d'une grammaire générique à une application spécifique reste un problème essentiel pour une utilisation pratique de ce formalisme dans des interfaces vocales. De plus expérimenter des algorithmes d'analyse et des heuristiques particulières nécessite des outils génériques et ouverts.

1.3 Méthodologie de recueil

1.3.1 Sous-langage d'un système de dialogue

En principe le système n'a pas à comprendre des mots hors du sous-langage. Ceci ne signifie pas un comportement déterministe impliquant l'arrêt d'une analyse en cas d'incomplétude de la grammaire ou d'agrammaticalité, ou encore un glissement d'un énoncé hors de la grammaire vers un énoncé grammatical par effet de bord par exemple d'une analyse probabiliste. Nous avons soutenu dans le chapitre précédent que l'usage d'un mot ou d'une structure hors de la grammaire doit être détecté en tant que tel et, si les tentatives de réparation échouent, entraîner des interactions verbales supplémentaires avec l'utilisateur.

1.3.2 Expérimentations de type Magicien d'Oz

Le corpus obtenu suite à une expérience de Magicien d'Oz, auquel on peut attribuer une représentativité subjective, devient alors une source de travail notamment pour la modélisation linguistique des énoncés en vue de la compréhension et de l'analyse du dialogue.

Un problème dans la conception de ce type de corpus réside dans la représentativité du recueil par rapport au sous-langage que l'on souhaite décrire. Si le principe de sous-langage est fondé, on peut considérer qu'à partir d'une certaine taille du corpus, l'augmentation de la taille du vocabulaire et de la grammaire n'évoluera plus significativement. Une méthodologie d'estimation de la taille du corpus à considérer pour atteindre une certaine représentativité est une chose *a priori* importante sinon nécessaire pour la mise en œuvre pratique d'expérimentations de Magicien d'Oz.

Notre démarche consiste à recueillir un corpus qui sera par la suite classiquement divisé en deux parties, la première permettant de concevoir une grammaire du sous-langage (*corpus de conception ou d'entraînement*) et la seconde dédiée aux tests (*corpus de tests*).

Nous avons évoqué différents aspects essentiels au type de système que nous mettons en œuvre : démarche expérimentale de type Magicien d'Oz afin de recueillir des corpus, spécialisation/conception d'une grammaire lexicalisée à partir de ces derniers dédiée à l'analyse de l'oral, test de la grammaire et évaluation de la représentativité du corpus de conception. Nous examinons ici quatre ateliers relativement à ces aspects.

1.4 Positionnement par rapport aux systèmes existants

Le système XTAG [Doran et al.94] propose une grammaire LTAG de l'anglais à large couverture, des modules de visualisation de la grammaire et de résultats d'analyse et un analyseur de type Earley [Schabes94]. Cependant trois principaux aspects nous semblent limitatifs :

- le système de conception de la grammaire est dédié à une grammaire générale de l'écrit et rien ne permet la restriction de cette grammaire à une sous-grammaire basée sur un corpus, ni une adaptation à l'analyse d'énoncés oraux ;
- l'analyseur fournit une réponse binaire (phrase acceptée ou refusée) difficilement compatible avec le test d'une grammaire de taille importante où on souhaiterait manipuler des résultats partiels et obtenir des diagnostics d'erreurs ;
- la conception d'un nouvel analyseur se justifie donc mais l'intégration au système XTAG de nouvelles composantes est hors de la compétence d'une personne n'ayant pas participé au développement du système. Plus généralement ce système n'a pas été conçu dans un souci de diffusion car il manipule des formats propres non spécifiés, nécessitant une expertise pointue pour son installation, etc.

L'atelier de « Génie linguistique » pour LFG du LIMSI [Briffault et al.97] couvre les mêmes possibilités que le système précédent avec cependant une grammaire plus modeste. Les points forts de ce système sont l'intégration d'un niveau sémantique basé sur les graphes conceptuels et des choix techniques réfléchis permettant son évolution comme son intégration dans d'autres systèmes.

Le système GEPETTO [Ciravegna et al.97] offre également un atelier de tests incluant plusieurs algorithmes d'analyse syntaxique. Ce système s'inscrit dans une démarche très générale, indépendante du formalisme. Cette généralité peut apparaître comme un avantage, mais entraîne une perte de fonctionnalité d'outils adaptés aux concepts et aux données d'un formalisme particulier. Si la méthodologie associée prévoit une division des corpus d'entraînement et de tests, il n'est pas prévu l'adaptation de ces outils dans d'autres applications, ni la spécialisation d'une grammaire générale.

Cette dernière constatation s'applique également pour l'environnement Hdrug [vN et al.97], outil utilisé à la fois pour le développement d'applications et pour l'expérimentation de stratégies d'analyse en paticulier pour l'oral. La grammaire utilisée est une grammaire HPSG simplifiée.

Nous allons maintenant présenter le système proposé, dont la spécificité par rapport aux ateliers existants est l'objectif de spécialisation grammaticale.

Chapitre 2

Spécifications et architecture du système EGAL

Nous présentons dans ce chapitre le fonctionnement du système EGAL, dont les objectifs ont été présentés au chapitre précédent. Ce système se compose d'un premier ensemble de modules assurant la gestion et la spécialisation des différentes ressources linguistiques associées au LTAG. Un second ensemble a pour tâche de tester la grammaire résultante par des tests d'analyse. Nous conclurons cette présentation par quelques détails sur l'implantation du système et une perspective d'extension à une TAG sémantique synchrone.

2.1 Modules de conception et de spécialisation

L'organisation générale d'une grammaire lexicalisée destinée à l'analyse syntaxique repose sur trois sources de données :

- une base morpho-syntaxique qui à une forme fléchie associe un lemme, une catégorie syntaxique et un ensemble de traits morphologiques ;
- une base syntaxique qui à un lemme donné associe un ensemble d'arbres élémentaires représentant les contextes syntaxiques dans lesquels ce lemme peut être utilisé ;
- enfin un ensemble de schèmes.

Ces trois types d'informations se combinent afin de produire les arbres élémentaires lexicalisés employés durant l'analyse.

La composante de conception grammaticale du système présenté ici s'articule autour de ces trois types de données. L'ensemble de l'architecture du système est représenté sur la figure 2.1.

Extraction morpho-syntaxique

L'extraction morpho-syntaxique consiste simplement à exploiter les bases de données morphologiques existantes (Multext, BDlex) en extrayant automatiquement les informations nécessaires. La liste des mots est dans un premier temps extraite du corpus d'entraînement, puis chaque mot fait l'objet d'un accès à BDlex ou Multext. Le résultat est un lexique morphologique correspondant au corpus d'entraînement qui associe à chaque entrée sa catégorie, son lemme et ses traits morphologiques.

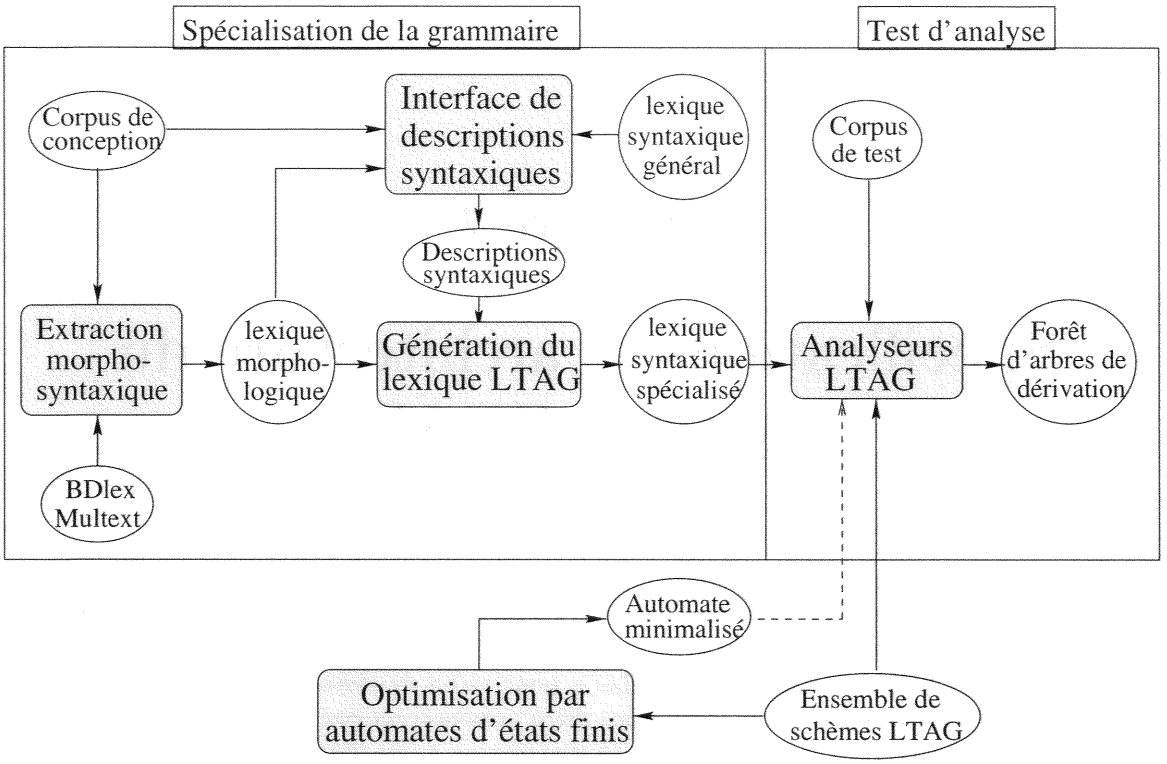


FIG. 2.1 – *Vue générale du système EGAL.*

Descriptions syntaxiques générales

L'objectif de ce module est d'identifier les propriétés syntaxiques à associer à un lemme afin de sélectionner les structures syntaxiques dans lesquelles il peut s'employer. Cette identification est la seule nécessairement non-automatisable. Elle se fait à l'aide d'une interface dédiée à des non-linguistes.

L'idée est de proposer à un utilisateur une série de tests linguistiques et de questions illustrés par des exemples afin de caractériser un lemme (par exemple : le verbe peut-il s'appliquer sous une forme réflexive? avec quel auxiliaire s'emploie-t'il?). Les réponses peuvent se faire soit de manière complète en vue de caractériser un lemme selon tous ses emplois possibles, soit en s'intégrant dans la méthodologie proposée, en vue de caractériser les seuls contextes apparaissant dans le corpus de conception. Une extraction des énoncés portant une occurrence du lemme est proposée simultanément (voir la vue d'écran (2.2)). Nous allons ici au-delà de la conception classique d'un sous-langage en spécialisant les lemmes en termes de catégories selectionnées mais également en termes de contextes syntaxiques spécifiques. Nous ne nous appuyons pas sur le principe de famille d'arbre introduit par le système XTAG, essentiellement pour des raisons de performances : plutôt que de reléguer la sélection des bons arbres d'une famille à associer à une ancre donnée par unification au moment de l'instanciation, les bons arbres sont déjà déterminés à l'aide de ces descriptions et seront directement notés dans le lexique syntaxique.

Nous considérons deux types de contextes syntaxiques :

- les contextes automatiques, c'est-à-dire qui amèneront automatiquement un certain

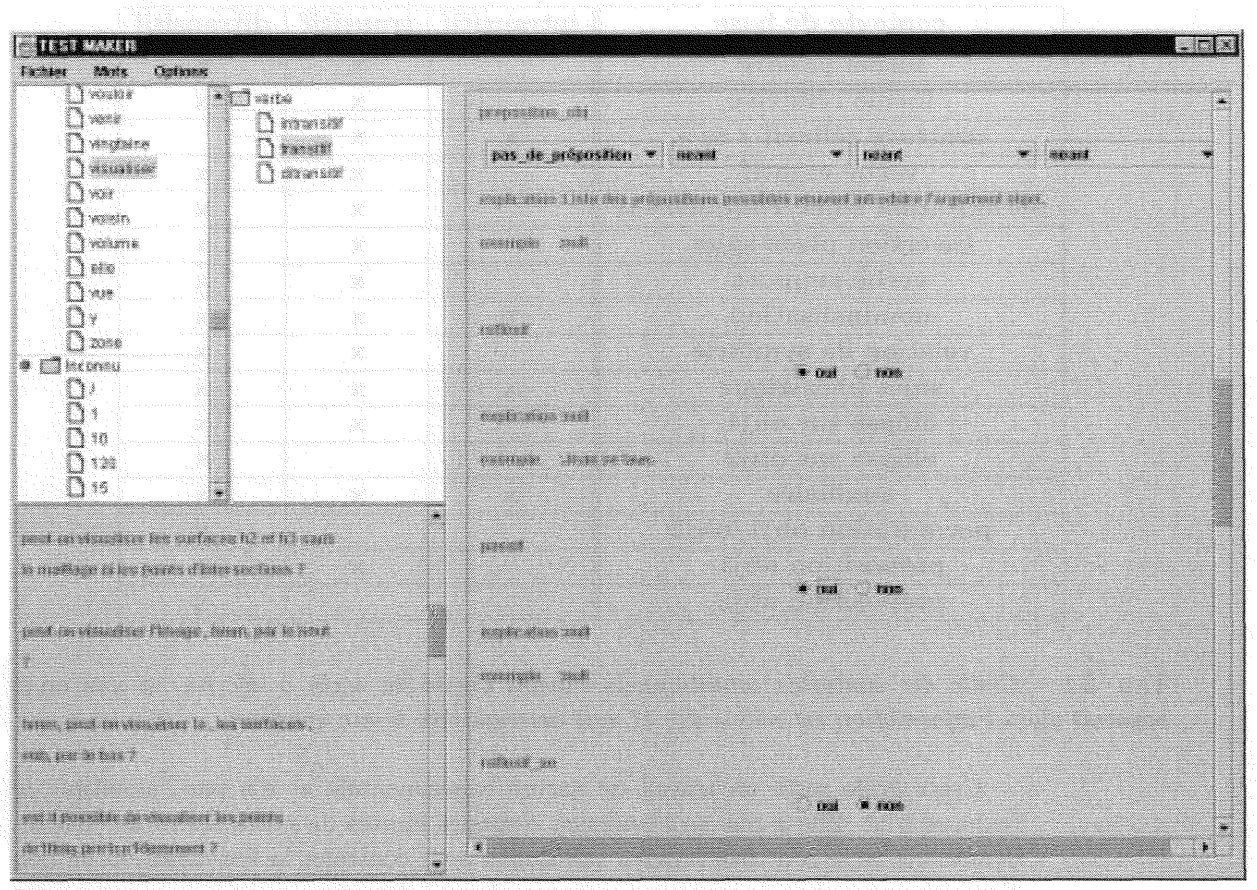


FIG. 2.2 – *Atelier de description assistée de lemmes.*

nombre d'arbres à être liés à un mot simplement sur la base de leur catégorie syntaxique (par exemple les déterminants) ;

- les contextes conditionnels, c'est-à-dire qui nécessitent des informations supplémentaires afin de décider si les arbres correspondant peuvent être ancrés par un mot particulier.

La table 2.1 donne pour les verbes la grille des questions à poser étant donné le contexte verbal de base considéré (ce dernier est déterminé par l'arité du verbe).

Cette grille de classification des verbes a été établie à partir de [SD96] et [Abeille91b] auxquels il est possible de se référer pour des descriptions des contextes syntaxiques du français. Chaque verbe se voit associé à une telle grille qui prend alors la forme d'une structure de traits sans variable.

contexte de base	intransitif	transitif	ditransitif
forme passive		x	x
forme réflexive		x	x
contexte réflexif « se »	x	x	
verbe ergatif		x	x
permutation sujet/complément		x	x
inversion place-sujet		x	x
verbe support		x	x
nominalisation	x	x	x
relation de symétrie		x	x
sujet phrastique	x	x	x
ellipse sur obj1		x	x
ellipse sur obj2			x
auxiliaire	x	x	x
permutation obj1/obj2			x
préposition obj1		x	x
préposition obj2			x

TAB. 2.1 – *Grille des contextes conditionnels suivant l'arité du verbe à décrire : si une croix apparaît alors l'information est nécessaire et doit correspondre à une question.*

Cette grille reste un travail de nature expérimentale et n'a pas l'ambition de l'exhaustivité. Elle a mis en l'évidence certaines limites de notre approche. Les informations ne se sont pas révélées suffisantes pour contrôler la lexicalisation ou non d'un certain nombre de contextes correspondants à des compositions de ces contextes conditionnels simples. Le travail de [Candito99], par contre, semble avoir identifié les informations nécessaires au contrôle des lexicalisations verbales de manière cohérente à une métagrammaire donnée. La méthodologie pourrait donc s'appliquer de façon identique avec cette nouvelle grille. Des exemples sont proposés pour illustrer l'emploi d'un contexte syntaxique particulier et aider à la décision.

Un outil complémentaire, à destination de linguistes cette fois, permet de concevoir les tests linguistiques. On peut noter que :

- ces descriptions sont indépendantes du formalisme lexicalisé qui est employé (LTAG ou HPSG par exemple) ;
- ce module permet d'intégrer de manière simple de nouveaux mots à un système. Une fois la catégorie et les traits morpho-syntaxiques pour ce nouveau mot définis, il suffit de le caractériser syntaxiquement en répondant aux questions soumises par cet outil.

Ensemble de schèmes

Nous partons du principe que nous disposons d'un ensemble de schèmes d'arbres élémentaires généré par exemple automatiquement à l'aide d'un système comme [Candito96] duquel notre système se veut complémentaire. La série de tests du module précédent revient en particulier à déterminer la place d'un lemme dans la structuration proposée dans [Candito99]. Notons qu'un éditeur permet de concevoir des schémas d'arbre nouveaux et de les intégrer au sein d'une arborescence de familles

d'arbre. Celui-ci est présenté sur la figure 2.3. D'autre part, un module d'optimisation par compaction d'automates d'états finis selon des techniques proches de [Evans et al.97] doit permettre une factorisation des sous-structures communes présentées par l'ensemble des schèmes afin de rendre plus efficace l'analyse (voir chapitre 4 de la troisième partie). Ce module d'optimisation n'est pour le moment pas encore implanté.

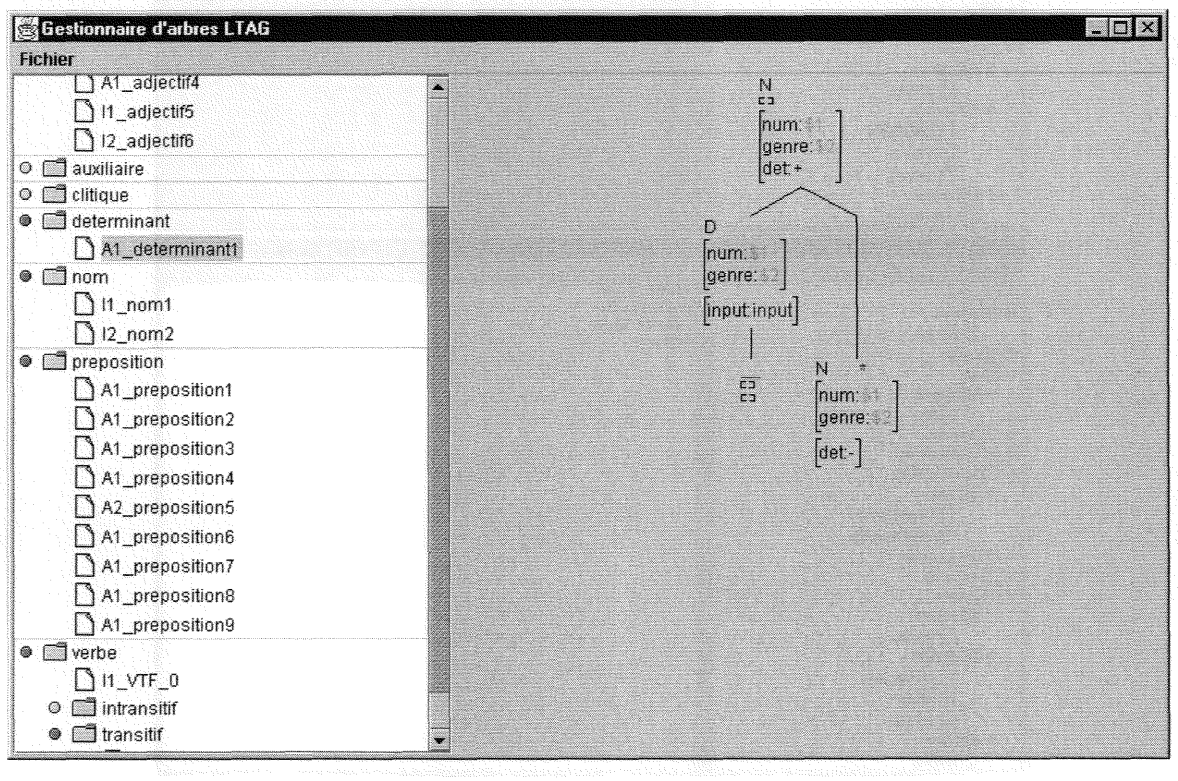


FIG. 2.3 – *Vue de l'éditeur de schèmes d'EGAL.*

Génération automatique du lexique syntaxique LTAG

Cette phase consiste à produire le lexique syntaxique spécialisé en exploitant les informations des trois bases de données précédentes. Les liens vers les schèmes sont simplement notés par références externes dans le fichier produit. Un fichier de ressources établit les correspondances entre critères de classification des descriptions syntaxiques et schèmes adéquats.

Ce résultat est un lexique syntaxique qui associe donc à une forme fléchie un ensemble de couple (lemme, traits morpho-syntaxiques), et pour chacun de ces couples un ou plusieurs liens vers un schème (avec les éventuelles co-ancres non instanciées). Un éditeur permet de visualiser ce résultat et d'effectuer les mises au point et corrections manuelles qui s'avèreraient nécessaires. L'éditeur est présenté sur la figure 2.4, il permet de visualiser les informations morphologiques et les schèmes référés.

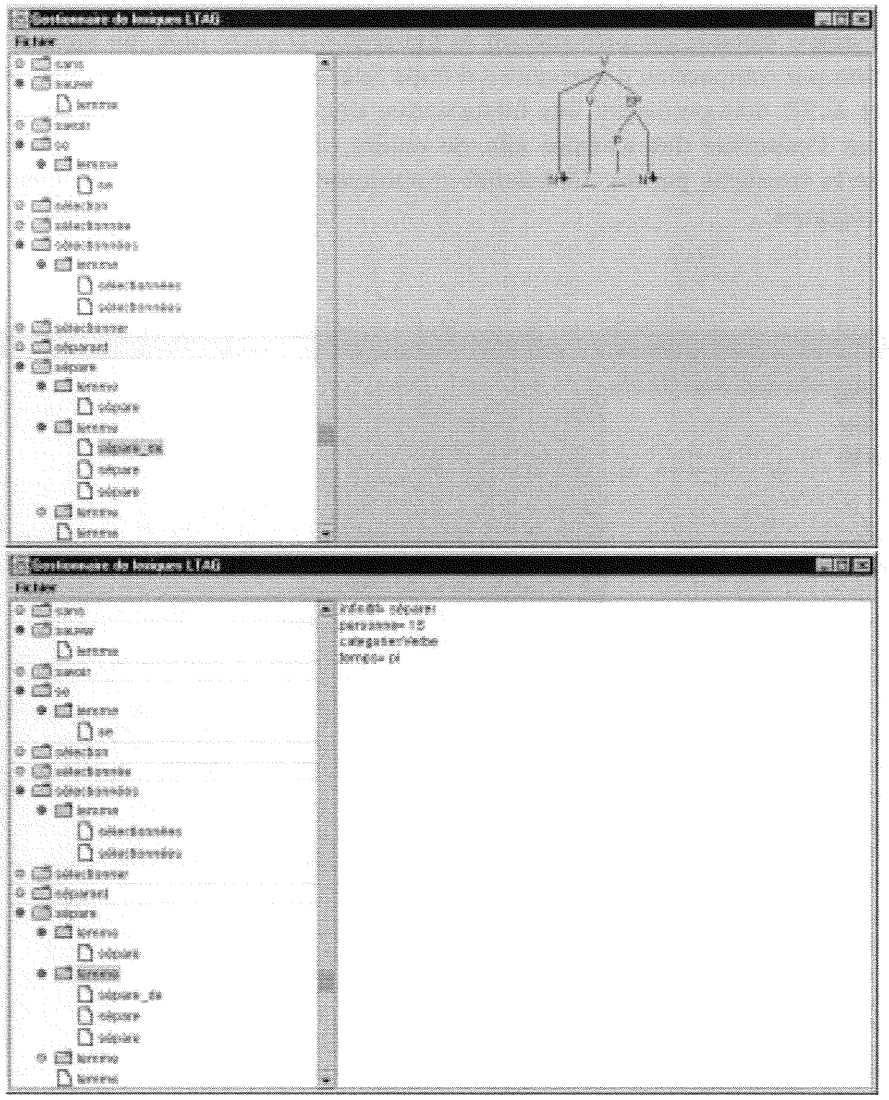


FIG. 2.4 – *Vue de l'éditeur du lexique syntaxique généré d'EGAL.*

2.2 Module de tests d'analyse

Après avoir obtenu une grammaire pour un sous-langage d'application à partir d'un premier corpus, le module de test d'analyse a pour tâche de tester le résultat obtenu sur un second corpus (voir vue 2.5). Ceci permet :

- de tester et comparer différentes heuristiques et stratégies d'analyse ;
- de visualiser[52] les résultats d'analyse complets et partiels afin d'améliorer la grammaire obtenue et d'étudier notamment les phénomènes agrammaticaux observés sur le corpus de test ;
- de donner des informations sur la représentativité du corpus initial qui a permis

52. Nous utilisons dans EGAL une API d'édition d'arbres conçue par Rodrigo Reyes (Thomson LCR et TALaNa)

de concevoir la grammaire.

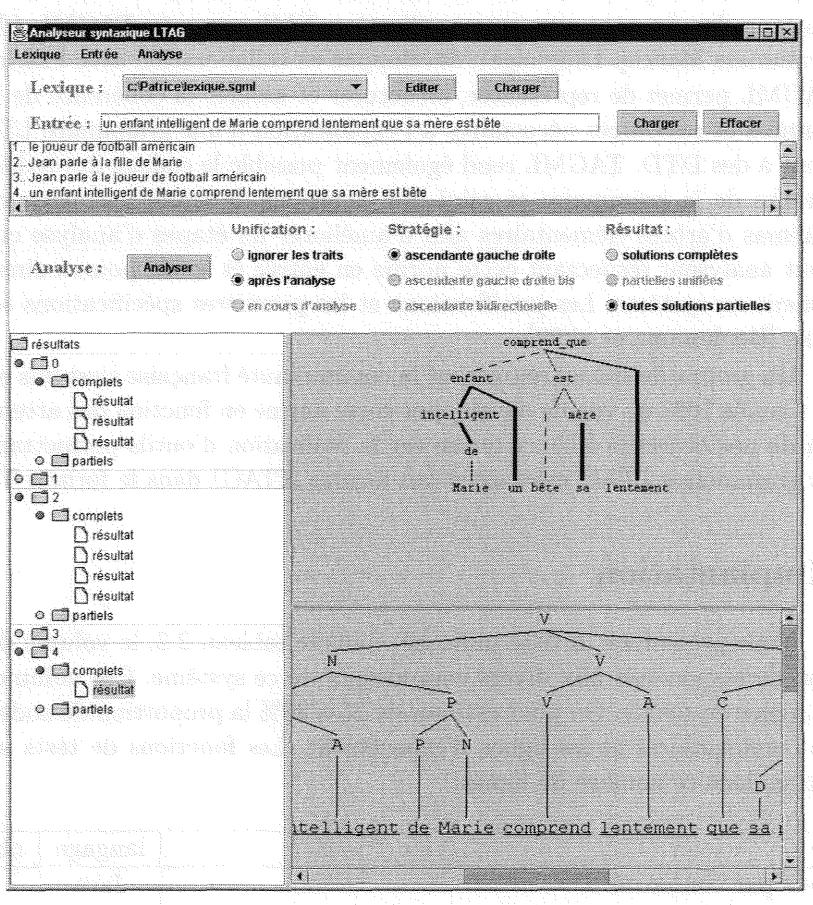


FIG. 2.5 – *Module de test d'analyse.*

Cet atelier propose plusieurs algorithmes et heuristiques d'analyse :

- un algorithme ascendant de type CKY pour les LTAG avec le mécanisme de prédiction simple suggéré dans [VS et al.93] ;
- un algorithme ascendant par connexité [Lopez98b] ;
- une implantation de l'algorithme descendant de [Schabes94] de type Earley, réalisée par Bertrand Gaiffe, est prévue d'être intégrée au système prochainement.

Les algorithmes ascendants fournissent les analyses complètes et partielles avec ou sans unification des structures de traits utilisées en LTAG. Ces différents types de résultats ont pour but de permettre un véritable test de la grammaire utilisée en identifiant l'étape impliquée dans un échec de l'analyse.

2.3 Choix techniques

L'ensemble de l'implantation s'appuie sur Java, essentiellement pour des raisons de portabilité qui faciliteront la diffusion de ces outils[53], et sur un codage systématique des données avec une application XML nommée TAGML (Tree Adjoining Grammars Markup Language), développée en collaboration avec Patrice Bonhomme. TAGML permet de représenter, structurer et assurer la cohérence de l'ensemble des données et ressources nécessaires à l'exploitation d'une grammaire LTAG conformément à des DTD. TAGML rend également possible le codage des redondances structurelles de la grammaire et celui des équations de traits partagées entre plusieurs schémas d'arbres élémentaires afin d'améliorer les étapes d'analyse et d'unification. Tout analyseur respectant cette norme en entrée et sortie pourra ainsi s'intégrer aisément au système. Les justifications et les premières spécifications sont présentées dans [Bonhomme et al.99].

Un groupe de travail réunissant la communauté française s'est mis en place à la fin de l'année 1998 en vue de développer cette norme en fonction des attentes de chacun. Un de nos objectifs à court terme est la réalisation d'outils permettant de convertir les grammaires LTAG existantes (au format XTAG) dans le format TAGML.

2.4 Implantation

Nous présentons, à titre indicatif, dans le tableau 2.2, le volume de programmation correspondant aux divers composants de ce système. Ces volumes sont donnés en lignes de source. On peut estimer de 25 à 30% la proportion de code employé pour les commentaires et les lignes d'espacement. Les fonctions de tests sont également inclus dans ce nombre de lignes.

module	langage	nb. lignes de sources
analyseur par connexité	Java	4936
éditeur de lexique morphologique, syntaxique et de schèmes	Java	1291
module d'extraction morphologique	Java	374
module de description syntaxique	Java	2043
module de génération de la grammaire LTAG	C++	940
atelier de test d'analyse	Java	2002
total		11586

TAB. 2.2 – *Tableau récapitulatif de l'implantation d'EGAL.*

Dans ces chiffres, nous n'avons pas compté les *packages* employés pour l'unifcation, l'édition d'arbres et l'analyse des documents XML. Nous remercions les contributions extérieures qui ont rendu possible la réalisation de ces différents outils :

- Rodrigo Reyès (TALaNa et Thomson LCR) pour l'éditeur générique d'arbres ;
- Bertrand Gaiffe (LORIA-CNRS) pour le package d'unification ;
- Patrice Bonhomme (LORIA-INRIA) pour son analyseur XML ;

53. Nous souhaitons rendre prochainement disponible l'ensemble des sources du système afin de susciter dans la communauté d'éventuelles contributions.

- Aurélien Samson (stagiaire de deuxième année ESIAL) qui a travaillé sur le module de description syntaxique.

2.5 Validation : grammaire et analyse du corpus GOCAD

Le corpus GOCAD a été divisé en corpus de conception (80% des énoncés) et de test (20% restant). Les données sur la grammaire LTAG finalement obtenue ont été présentées dans le chapitre 3 de la $3^{ème}$ partie de ce mémoire. Le nombre total de liens vers un schème donne une métrique de la taille totale du lexique syntaxique.

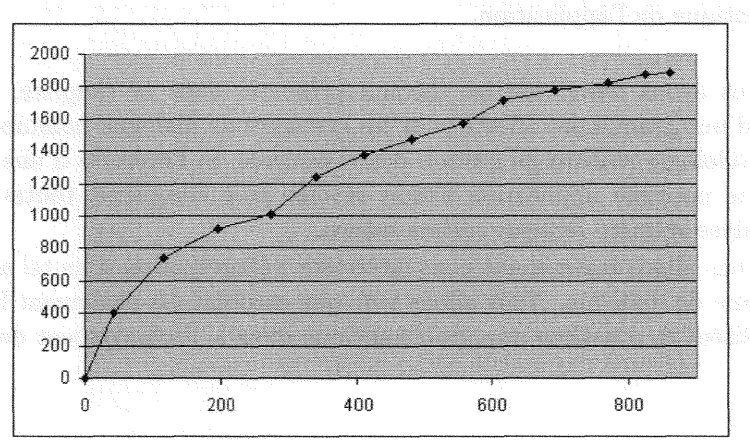


FIG. 2.6 – *Evolution de la taille de la grammaire LTAG générée (en nombre de liens vers un schéme) en fonction de la taille du corpus (en nombre d'énoncés).*

Les phases d'extraction et de génération du lexique syntaxique pour Gocad étant d'une durée faible (de l'ordre de 10 à 20 secondes pour chacune d'elles), il est possible de réaliser des tests systématiques sur l'évolution des données générées. La méthode proposée consiste dans un premier temps à faire des tirages aléatoires d'énoncés et de construire automatiquement la grammaire LTAG associée à l'aide du système EGAL. Ceci permet d'étudier l'évolution de la taille de la grammaire LTAG (donnée par le nombre total de liens vers un schème) en fonction de l'augmentation du nombre d'énoncés pris en compte. Ainsi, plus on se rapproche d'une asymptote horizontale plus la couverture du sous-langage est bonne. La figure 2.6 donne l'évolution observée pour le corpus GOCAD. On peut constater que, malgré une stabilisation de la courbe, un volume d'entraînement quelque peu supérieur est nécessaire pour aboutir à une bonne représentativité selon ce critère.

2.6 Vers le système EGALS : spécialisation et mise en œuvre de grammaires synchrones

La perspective naturelle d'une telle architecture est une évolution vers la prise en compte de la sémantique prédicative (syntaxe profonde) à l'aide de TAG synchrones [Shieber94]. L'intégration des contraintes sémantiques dans le processus d'analyse

sera permis par la synchronisation, et reposera sur le développement des deux points suivants :

- l'adaptation des algorithmes d'analyse pour la prise en compte des synchronisations dynamiques ;
- l'adaptation des outils de conception et de description en vue de définir la synchronisation des données statiques : en particulier tous les membres d'une même famille d'arbres sont synchrones à une forme normale prédicative, il faut synchroniser également lexèmes et sèmes.

Il sera alors possible d'étendre la spécialisation des données à tout le niveau linguistique et d'étudier l'injection au cours de ce processus de contraintes liées à la sémantique de l'application.

Nous avons tenté, par les travaux présentés dans ce chapitre, d'assoir l'utilisation d'une grammaire LTAG pour les systèmes de dialogue homme-machine sur une méthodologie réaliste de gestion des ressources. Si l'écriture d'une grammaire pour chaque nouvelle application s'était révélée trop complexe, toutes les propositions d'analyse robuste seraient restées vaines.

Nous allons maintenant nous intéresser à l'intégration d'un tel analyseur dans un système de dialogue. Nous allons voir que ce point est également lié aux ressources mobilisées et constitue une problématique dans le prolongement du système EGAL.

Chapitre 3

Intégration des traitements fondée sur la synchronicité

3.1 Analyseurs synchrones au sein d'un système de DHM

Nous étudions dans ce chapitre la place et l'utilisation de l'analyseur réalisé pour un système complet et son intégration avec le processus de reconnaissance de la parole. Cette réflexion a été menée en collaboration avec Jean-Luc Husson (LORIA), dont le domaine d'étude est le traitement automatique de la parole et plus spécifiquement la segmentation du signal acoustique. Notons que les principes proposés ici restent un travail préliminaire, l'objectif étant de nous doter de moyens d'expérimenter des mécanismes de coopération entre reconnaissance de la parole et syntaxe.

Nous proposons et justifions les bases d'une architecture permettant une forte intégration des différentes contraintes d'analyse. Notre ambition est de bénéficier de l'efficacité des modèles probabilistes et d'en limiter les biais pour l'interprétation en offrant la possibilité d'exploiter explicitement des connaissances expertes. Cette intégration repose sur l'homogénéité des structures manipulées par les niveaux de traitement, des automates synchronisés à partir desquels les différents modules créent des états temporaires de recherche, eux aussi synchronisés en conséquence. La manipulation de ces états temporaires est conditionnée par des contraintes propres aux mécanismes du niveau local de traitement (symboliques et/ou probabilistes) et par des contraintes globales impliquées par la synchronisation deux à deux des modules.

Ces différents principes ont pour objectif d'expérimenter des stratégies d'intégration d'un système de segmentation analytique de la parole continue, d'un module stochastique de reconnaissance de phonèmes et de l'analyseur LTAG présenté dans les parties précédentes.

3.2 Les systèmes existants : avantages et limites

3.2.1 Architecture classique

Nous présentons figure 3.1 l'architecture classique adoptée pour les composantes chargées de la reconnaissance et de l'analyse syntaxique des énoncés. Un HMM (*Hidden Markov Model*) réalise le décodage acoustico-phonétique du signal. La combinatoire de reconnaissance est contrôlée dynamiquement par un modèle de langage

de type *N-gram* (bi ou trigramme). L'intégration de ces deux traitements s'effectue exclusivement par la combinaison des probabilités. Le résultat de ce niveau de traitement infra-lexical est alors un treillis pondéré des hypothèses de mots, qui sera, dans un second temps, exploité par un module d'analyse syntaxique, le plus souvent grâce à une grammaire hors contexte ou à une analyse par segments [Abney91]. L'intégration dans ces architectures repose donc sur deux techniques complémentaires, l'intégration par treillis et celle par combinaison de probabilités.

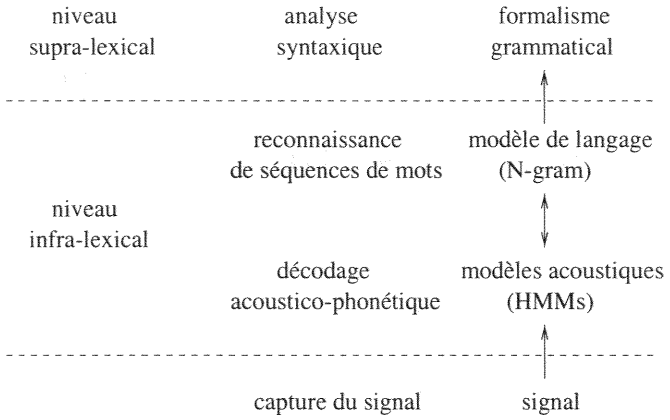


FIG. 3.1 – *Composantes de reconnaissance et d'analyse syntaxique pour les architectures « classiques ».*

3.2.2 Intégration par « treillis »

Une approche par treillis permet aux composantes syntaxique et sémantique de n'explorer que les meilleures hypothèses de reconnaissance, mais elle dissocie les composantes de reconnaissance et de structuration des énoncés, ce qui ne va pas sans poser un certain nombre de problèmes. Tout d'abord, le treillis est construit à partir des contraintes d'ordre des mots du modèle de langage. Cette technique est bien adaptée pour capturer des contraintes locales, mais inefficace pour capturer des relations hiérarchiques, même simples, comme les accords sujet-verbe ou les rattachements prépositionnels particulièrement fréquents pour les dialogues de commande qui nous intéressent [54]. De plus, les modèles de type N-gram acceptent des hypothèses d'énoncés agrammaticaux qu'on ne saura de toute façon pas interpréter et qu'il sera nécessaire de filtrer *a posteriori* [Roussel et al.97]. Enfin le système de reconnaissance devant évaluer chaque mot à chaque point du signal comme une hypothèse séparée [Price et al.89], cette approche est de surcroît coûteuse. Ces observations justifient l'intérêt d'une grammaire linguistiquement motivée qui permet notamment d'exprimer des contraintes de dépendance à distance. Il est important de noter que l'utilisation d'une telle grammaire linguistiquement motivée n'est pas incompatible avec celle de statistiques (grammaires stochastiques), offrant alors, en se substituant à un modèle *N-gram* de langage, des possibilités intéressantes de diminution de la

54. Notons néanmoins que ce résultat convient bien pour des approches où de toute façon l'interprétation sera obtenue par des méthodes sélectives sur des segments des hypothèses d'énoncés reconnus.

perplexité de l'analyse. De bons résultats de perplexité ont été obtenus par exemple par [Srinivas97a] à l'aide d'un modèle de langage fondé sur un *supertagging* à partir d'une grammaire LTAG de l'anglais.

3.2.3 Intégration « tout stochastique »

Les bons résultats atteints par les systèmes stochastiques de reconnaissance (HMMs) sont dus à la rigueur de leurs fondements théoriques qui régissent les phases d'apprentissage et de reconnaissance. Cependant, il semble que le seul moyen pour améliorer l'aptitude des HMMs à capter la grande variabilité du signal soit d'augmenter continuellement le nombre de paramètres des modèles (utilisation de multigaussiennes, de modèles d'ordre 2, etc.) et parallèlement la taille des corpus indispensables à leur apprentissage. La stagnation constatée des performances des HMMs s'explique, à notre avis, par la grande difficulté rencontrée pour intégrer de manière explicite des connaissances acoustiques, liées à la production ou à la perception de parole. Quelques connaissances peuvent être prises en compte lorsqu'elles ne remettent pas en cause le cadre d'application des algorithmes d'apprentissage et de reconnaissance. Des informations perceptives simples ont pu ainsi être prises en compte en utilisant des techniques de paramétrisation de type PLP[55], MFCC, qui utilisent une échelle perceptive non linéaire (Mel), ou issues d'un modèle auditif [Seneff86]. Des coefficients dérivés du premier et du second ordre sont ajoutés aux vecteurs de paramètres pour tenir compte de la forte corrélation acoustique observée sur certaines parties du signal et pour compenser les effets de l'hypothèse forte d'indépendance des observations successives. L'introduction de contraintes de durée des phonèmes est très complexe et se réduit généralement à adapter empiriquement la topologie des modèles au phonème considéré. La prise en compte de l'évolution dynamique de cette durée en fonction de la place du mot dans le syntagme et du rythme d'élocution est impossible à ce niveau. Malgré les performances atteintes par les HMMs, il semble que les contraintes structurelles (types de modèles, algorithmes) imposent *a priori* des limitations intrinsèques aux améliorations possibles.

La combinaison de probabilités constitue une solution relativement simple pour intégrer des informations hétérogènes. Se baser également sur un modèle de grammaire stochastique ou DOP[56] pour mener l'analyse syntaxique permettrait alors une intégration directe des diverses contraintes statistiques des différents niveaux de traitement. Cependant la pertinence d'un énoncé ne peut se réduire à des critères statistiques et il nous semble nécessaire de compléter l'efficacité prédictive des trigrammes par des constructions résultant d'une expertise linguistique, capable d'apporter des éléments de décision sur des critères d'analyse autres que de simples considérations liées à la fréquence d'observation. À titre d'exemple, [Roussel et al.98a] propose un modèle de décision multicritère floue pour déterminer le meilleur compromis entre critères de haut niveau et reconnaissance stochastique. Cependant l'évaluation a montré que la classification des hypothèses nécessite des critères de haut niveau (identification de l'acte de dialogue et la pertinence des expressions referientielles). Enfin contrairement à ce que sous-entendrait une intégration stochastique, les rapports entre deux niveaux de traitement ne sont pas probabilistes, il s'agit d'informations statiques qu'il convient de représenter explicitement.

55. *Perceptually-based Linear Prediction* [Hermansky90].
56. Data Oriented Parsing, voir [Bod95]

3.2.4 Homogénéisation des connaissances et automates

L'objectif d'intégration que nous souhaitons développer suppose une structure de représentation des connaissances commune à tous les modules hétérogènes. Nous considérons que les automates d'états finis sont particulièrement bien adaptés pour les raisons suivantes :

- l'utilisation d'automates d'états finis repose sur des techniques bien maîtrisées, simples à implanter et efficaces ;
- il est possible de représenter des structures complexes par des automates et de mener des recherches sur ces structures de données ;
- les automates pondérés permettent d'intégrer des considérations probabilistes [Pereira et al.91] ;
- les automates obtenus peuvent être minimalisés de façon à obtenir un partage optimal des sous-structures communes.

Notre hypothèse en particulier est que cette propriété de partage peut conduire à un taux important de factorisation des calculs d'analyse. Les automates d'états finis sont utilisés couramment en reconnaissance de la parole : la représentation de mots sous la forme d'un graphe de phonèmes permet une représentation compacte sur laquelle mener une reconnaissance. Les travaux du LADL ont par exemple démontré l'efficacité des techniques de minimalisation d'automates en lexicologie pour une tâche d'analyse et de génération de mots dérivés[57]. Nous allons maintenant présenter comment les différents niveaux de traitement peuvent être appréhendés comme des processus fonctionnant sur des données statiques représentées, malgré leur hétérogénéité, par des structures uniformes d'automates.

3.3 Automates de recherche et niveaux de traitement

3.3.1 Segmentation

Présentation du système

Le module de segmentation utilisé repose sur la technique des réseaux de segmentation multiniveaux [Zue et al.89] qui permet de représenter l'ensemble des segmentations possibles du signal (de la plus fine à la plus grossière) dans une structure hiérarchique uniforme appelée *dendrogramme*.

L'algorithme présenté dans [Husson et al.96] permet de retrouver dans cette structure hautement combinatoire le sous-ensemble des solutions de segmentation phonétique les plus vraisemblables. L'efficacité de cet algorithme est due à l'intégration explicite de connaissances acoustiques, en particulier, un modèle de durée des phonèmes, un modèle d'homogénéité spectrale, ainsi que diverses contraintes telles que le respect des frontières de voisement détectées et l'estimation sous la forme d'un intervalle de confiance du nombre de segments acoustiques probablement présents sur le signal en fonction de sa durée observée et du mode de voisement associé. Ces critères et ces contraintes sont générés dynamiquement et combinés analytiquement

57. À titre d'exemple, l'automate qui permet la reconnaissance de 1.301.976 entrées n'occupe que 930 Ko en mémoire centrale et procède à hauteur de 15.000 mots à la seconde sur un ordinateur NeXT [Clemenceau93].

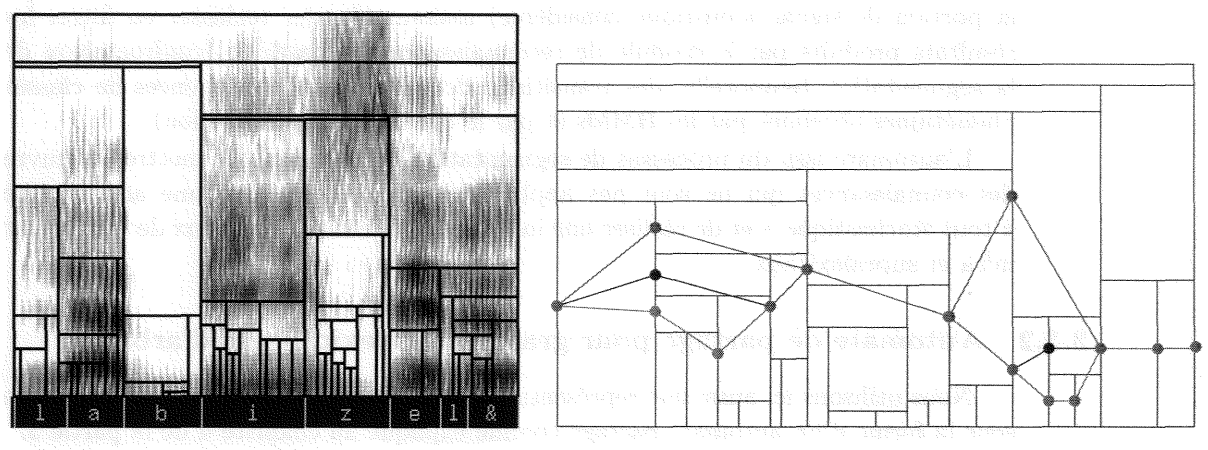


FIG. 3.2 – *Exemples de dendrogramme réel et d'automate de solutions de segmentation sur un dendrogramme stylisé.*

pour fournir un ensemble réduit des n hypothèses les plus vraisemblables. Les résultats d'évaluation placent ce système en bonne place par rapport aux outils de segmentation phonétique concurrents [Husson99].

Le système calcule incrémentalement durant le parcours du dendrogramme l'automate déterministe minimalisé représentant l'ensemble courant des segmentations les plus vraisemblables. Les états correspondent à des frontières temporelles entre des sons successifs. Les transitions sont étiquetées d'informations issues du processus de segmentation (durée calculée, macro-classe phonétique estimée). Un exemple d'automate minimalisé produit est fourni figure 3.2. Sur cet exemple, la minimalisation permet de passer de 30 à 16 états.

Intérêts de l'utilisation de ce module

Interaction avec le module d'analyse syntaxique. Les interactions avec le niveau de traitement supralexical s'effectuent par l'intermédiaire d'un automate phonétique lui même contraint par un module d'analyse syntaxique utilisé en mode prédictif. Ce graphe phonétique permet d'affiner certaines contraintes exploitées par la segmentation comme le nombre de segments supposés présents, les séquences de classes phonétiques ou les transitions de voisement attendues. Réciproquement, le module syntaxique étant capable de situer le mot (et donc les sons correspondants) dans le groupe rythmique, ces informations peuvent également être propagées à l'automate phonétique et donc, dans un second temps, utilisées pour ajuster les paramètres des modèles de durée utilisés par la segmentation. Parallèlement, les résultats de segmentation obtenus indépendamment de ces contraintes spécifiques permettent de classer voire de filtrer, sur la base d'indices acoustiques mesurés, certaines hypothèses d'analyse syntaxique.

Interaction avec le module de reconnaissance stochastique. Notre objectif n'est pas de contraindre la reconnaissance par HMMs par une segmentation *a priori* mais d'exploiter les hypothèses segmentales tout d'abord abductivement pour contrôler les modèles phonétiques appliqués à un moment du traitement de reconnaissance (selon la macro-classe phonétique ou le mode de voisement calculés pour

la portion de signal acoustique considérée) mais aussi pour reclasser ou filtrer les résultats produits par le module de reconnaissance stochastique (confrontation de la segmentation temporelle, des transitions de voisement, les séquences de classes phonétiques obtenues par les HMMs et par le module de segmentation).

L'automate issu du processus de segmentation permet donc de mettre en œuvre des connaissances qui ne sont pas applicables directement dans une architecture « tout stochastique » et de réaliser une interface riche entre les niveaux de traitement infra et supralexicaux.

3.3.2 Automate de partage pour grammaires lexicalisées d'arbres

Nous utilisons ici aussi une représentation de l'ensemble de la grammaire utilisée sous la forme d'un automate partagé comme expliqué au chapitre 4 de la partie 3.

3.4 Intégrations d'hypothèses par automates

3.4.1 Synchronisation d'automates

L'idée de cascades d'automates d'états finis et de combinaison de transducteurs pour la réalisation d'analyses sur plusieurs niveaux de traitement a été exploitée dans différents travaux, par exemple [Abney91] et [AM et al.97]. La synchronisation quant à elle est une technique également largement éprouvée pour la traduction automatique. Les transducteurs d'états finis permettant des traductions entre langages réguliers ont trouvé des applications notamment en analyses morphologique et phonologique. Les *PushDown transducers* (PDT) réalisent des traductions entre langages hors contextes. La synchronisation des Grammaires d'Arbres adjoints (TAG) a été proposée initialement dans [Shieber et al.90].

La définition utilisée ici de points de synchronisation entre deux automates ne repose pas sur la notion de transducteur, dans la mesure où nous ne souhaitons pas produire des traductions entre langages réguliers mais simplement établir des correspondances entre structures linéarisées se présentant sous la forme d'un langage régulier. De plus, notre objectif est d'une part d'éviter de dupliquer les structures au niveau des arcs de transitions, comme pour un transducteur, et d'autre part de ne pas combiner des automates qui relèvent de processus de traitements différents. La définition est la suivante : la *synchronisation* d'automates S est définie par le triplet (A_1,A_2,d_S) où A_1 et A_2 sont les automates synchronisés et d_S la fonction de synchronisation qui associe un ou plusieurs états de A_1 à un unique état de A_2. Nous disons que la synchronisation entre A_1 et A_2 est assurée si pour chaque état σ_1 reconnu de A_1 son état synchronisé $\sigma_2 = d_S(\sigma_1)$ est également reconnu et que la réciproque est également vraie.

Au cours de l'analyse, en fonction des données à analyser, les modules de traitement initialiseront des états temporaires liés à au moins un état de ces automates. Par conséquent, la synchronisation est de fait étendue aux états d'analyse.

3.4.2 Synchronisation des traitements de reconnaissance

Le principe de synchronisation mis en œuvre peut être illustré grâce à la figure 3.3. Une hypothèse de segmentation correspond à un chemin dans l'automate segmental

présenté figure 3.2. La fonction de synchronisation avec l'automate phonétique est construite dynamiquement avec l'automate segmental. On obtient alors un lien entre transitions correspondant à une macro-classe phonétique et les phonèmes possibles distribués dans le vocabulaire des corpus. Ainsi une hypothèse d'une segmentation à valider au niveau phonétique amènera à des hypothèses de mots, et inversemment une hypothèse de mot suppose un certain chemin dans l'automate de segmentation. La même démarche est appliquée avec l'automate syntaxique : une hypothèse de mot correspond à un chemin dans l'automate de phonèmes et donc à une hypothèse d'ancre lexicale pour le niveau syntaxique.

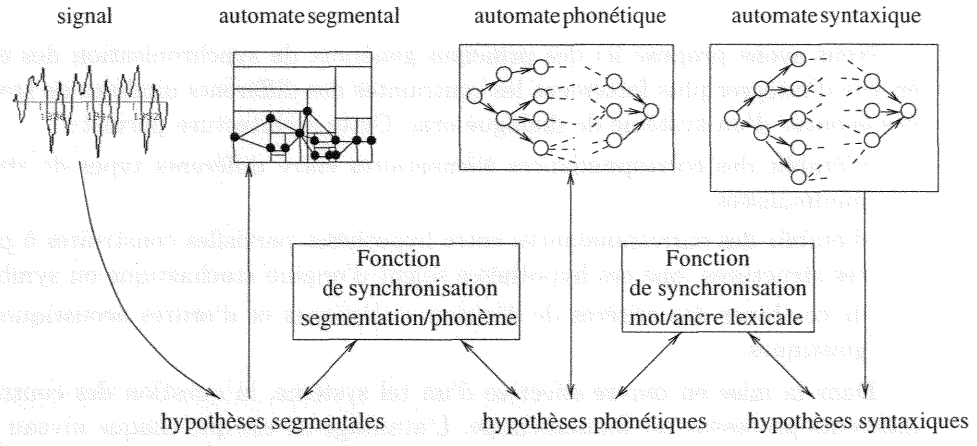


FIG. 3.3 – *Principes de synchronisation des composantes RAP et analyse syntaxique.*

Des structures temporaires sont créées par chaque module à partir de l'automate associé pour représenter les hypothèses concurrentes. L'indéterminisme de chaque niveau justifie notre stratégie de ne pas filtrer directement des hypothèses d'analyse indépendamment des autres contraintes, comme c'est généralement le cas dans les systèmes existants. Aussi privilégions-nous les techniques tabulaires, couramment utilisées pour les traitements symboliques non-déterministes de l'écrit.

3.4.3 Génération des automates synchronisés

L'extraction des automates synchronisés peut être obtenue à l'aide du système EGAL, qui s'appuie lui-même sur le lexique BDLEX. La liste de mots extraite d'un corpus et les transcriptions phonétiques de BDLEX nous permettent de réaliser l'automate phonétique sur lequel se base le module de reconnaissance. Par ailleurs, de façon semi-automatique comme expliqué dans le chapitre précédent, nous extrayons la grammaire syntaxique de l'application dont nous synchronisons les reconnaissances des ancres sur celles, dans l'automate phonétique, des mots pouvant ancrer l'arbre élémentaire.

3.5 Liens avec les niveaux supérieurs de compréhension

Dans [Shieber et al.90] est introduite la notion de synchronicité entre grammaires TAG notamment pour l'interface entre syntaxe et sémantique. Les synchronisations entre arbres élémentaires d'une grammaire lexicalisée d'arbres syntaxiques et d'arbres sémantiques peuvent aussi se traduire dans deux automates minimalisés synchronisés[58]. D'autre part, un certain nombre de travaux récents tentent d'exploiter l'idée de grammaire lexicalisée d'arbres pour représenter la structure du discours, par exemple [Webber et al.98]. Les travaux réalisés ici pourraient alors être également étendus au niveau discursif.

3.6 Bilan

Nous avons proposé ici des principes généraux de synchronisation des données, en vue d'intégrer plus fortement les contraintes des différents modules de traitement des énoncés d'un système de dialogue oral. Cette architecture permet :

- d'établir des correspondances élémentaires entre différents types de structures minimalisées ;
- d'établir des correspondances entre hypothèses partielles construites à partir de ces structures, que ces hypothèses soient d'origine stochastique ou symbolique ;
- de combiner des critères de décision statistiques et d'ordres acoustiques et linguistiques.

Dans la mise en œuvre effective d'un tel système, la question des contrôles des différents processus est fondamentale. L'avantage ici est que chaque niveau est susceptible de diriger une analyse, émettant des hypothèses valides selon le processus de traitement local, qui seront propagées aux autres niveaux de traitement chargés à leur tour de confirmer ou non ces hypothèses. Une architecture de type asynchrone, fondée sur un système multi-agents, nous semble le plus à même d'exploiter les propositions faites ici. L'architecture non centralisée destinée aux systèmes de dialogue, proposée dans [Lopez96], nous semble être tout à fait adaptée à nos besoins.

58. Notons ici cependant qu'en cas de reconnaissance partielle d'un automate, la synchronisation porte sur l'ensemble partiel des états synchrones du second automate qui peuvent ne pas être contigüs et donc faire perdre à ce deuxième niveau d'analyse le bénéfice de techniques tabulaires (analyse de complexité polynomiale au pire des cas devenant exponentielle au pire des cas). Cependant ce problème est général de l'interfaçage entre analyse syntaxique (généralement polynomiale, voire linéaire) et sémantique (potentiellement exponentielle car nécessitant une énumération des dérivations).

Conclusion et Perspectives

1 Bilan

Bien des critiques ont été émises à l'encontre de la linguistique et de l'informatique sur la rigidité des concepts et des modélisations que ces deux disciplines véhiculaient. À notre sens, la rencontre et l'avenir commun de ces deux domaines passent par une meilleure compréhension de la complexité des comportements et connaissances humaines sur le plan langagier. Le risque est de nous imposer, pour la linguistique, des schémas d'expression, voire de pensées, trop rigides et pour l'informatique des formatages de nos comportements interactionnels, un apauvrissement de l'utilisation de notre propre langue et une dépendance toujours plus grande aux informations. Linguistique comme informatique se doivent de permettre aux technologies à venir d'offrir plus de liberté à leurs utilisateurs. C'est en tout cas le fondement moral sur lequel nous avons tâché de réaliser ce travail de thèse.

Ainsi dans le domaine où s'inscrit cette thèse, si on souhaite permettre la mise en œuvre de dialogues homme-machine naturels et accessibles au plus grand nombre, il faut appréhender toute la variabilité et la complexité des phénomènes langagiers. Les propositions développées se fondent sur les travaux de Claire Blanche-Benveniste et du GARS sur le français parlé. Un premier point essentiel est le rejet de critère d'acceptabilité autre que celui des observations dans des corpus d'énoncés oraux. Le second point est que la syntaxe de la langue orale n'y est pas vue de façon autonome à celle de l'écrit : les énoncés y sont perçus comme se développant suivant deux axes, le premier, l'axe syntagmatique, correspond à un enchaînement grammatical des syntagmes, tandis que sur le second, l'axe paradigmatique, s'effectue la réalisation des syntagmes avec toute la variabilité due aux phénomènes oraux comme les hésitations, les corrections et les reprises. De ce modèle descriptif, nous avons déduit les bases d'un modèle cette fois procédural du français parlé : une première étape vise à mener une analyse très proche des normes de l'écrit (aux phénomènes d'effacement et d'ellipses près) afin d'isoler les segments grammaticaux les plus étendus. Il s'agit en fait directement de développements suivant l'axe syntagmatique. Un second niveau d'analyse couvre les variations paradigmatiques, localisées par un arrêt de l'analyse à l'étape précédente. Il a pour but de rétablir l'axe syntagmatique avec des traitements spécifiques aux types de phénomènes observés. Ces deux niveaux d'analyse s'alternent jusqu'à ce que l'on obtienne, par extension suivant l'axe syntagmatique, les îlots les plus étendus possibles, c'est-à-dire les plus riches pour la suite de l'interprétation.

En ce sens, nous avons proposé tout d'abord un analyseur syntaxique qui tente de tirer le meilleur parti des différentes théories et techniques informatiques modernes. Chacune de ces

techniques a connu de nombreux succès pour le Traitement Automatique des Langues, ceci grâce soit à une grande qualité descriptive de la langue, soit à une efficacité informatique particulière :

- une Grammaire Lexicalisée d'Arbres Adjoints apte à capter des phénomènes linguistiques complexes et à localiser les dépendances sémantiques ;
- des techniques tabulaires afin de mener une analyse performante tout en tenant compte du non-déterminisme de la langue ;
- un invariant d'analyse original permettant d'obtenir les îlots grammaticalement corrects les plus étendus, résultats particulièrement robustes et plutôt caractéristiques d'analyseurs par segments. Cet invariant nous permet en particulier de localiser plus précisément les variations paradigmatiques que les propositions déjà existantes ;
- des techniques à états finis pour une pré-compilation efficace de la grammaire.

Le choix du formalisme linguistique, dans notre cas les Grammaires Lexicalisées d'Arbres Adjoints, est très important puisque de là découlent de nombreuses possibilités, tant théoriques que techniques. Nous avons tenté de justifier l'ensemble de ces choix au vu de notre objectif initial, de travaux linguistiques sur la langue parlée et sur des contraintes d'efficacité des traitements informatisés imposées par les systèmes de dialogue.

Afin d'adapter plus particulièrement cet analyseur à l'analyse de la langue orale observée dans le dialogue homme-machine, nous avons étudié trois extensions de ces techniques d'analyse :

- une extension des Grammaires d'Arbres Adjoints afin de représenter plus finement certaines contraintes sur les énoncés incomplets. Nos propositions ne changent pas les principes existants du formalisme et permettent d'exploiter le potentiel prédictif des structures de traits ;
- une extension naturelle de l'analyse tabulaire proposée afin de capter des phénomènes agrammaticaux liés à l'oral et les énumérations : les règles paradigmatiques additionnelles ;
- une extension fondée sur des techniques d'états finis et les TAG synchrones en vue de synchroniser plusieurs niveaux de traitement d'un système de dialogue.

La grammaire LTAG est donc vue dans notre travail comme une structure modélisant les contraintes syntaxiques de l'axe syntagmatique, tandis que les règles additionnelles ont pour rôle de rétablir le déroulement syntagmatique brisé par des variations paradigmatiques liées à l'oralité. Ainsi, dans ce cadre, l'arrêt de l'analyse standard basé sur la grammaire LTAG permet de localiser les variations paradigmatiques, qui sont neutralisées par les règles additionnelles dont l'application est monotone et intégrée à l'algorithme tabulaire employé.

Nous avons dans cette thèse reconsidéré la place de la grammaticalité dans la syntaxe. Les formalismes syntaxiques ont toujours été conçus pour modéliser la compétence linguistique et les structures que l'on juge correcte. Leur ambition commune était la génération de toutes les phrases d'une langue donnée [Chomsky64]. Ces deux objectifs se révèlent contradictoires pour l'oral, où les structures syntaxiques employées présentent une grande variabilité. La solution que nous proposons prend en compte ces différents aspects, en acceptant les limites des formalismes face à la langue orale. Cette variabilité de l'oral dépend de nombreux facteurs et se réalise dans ce que l'on appelle souvent un niveau de langue (langue plus ou moins soutenu, relaché, fragmentaire, etc.). L'un des facteurs principaux

de cette variabilité, par exemple en dialogue homme-machine, est la fonction sociale que l'on projète dans la machine.

Ce travail n'est pas un aboutissement : il ouvre, à notre sens, de nombreuses perspectives qui pourront s'appuyer sur les différents outils développés pour l'atelier EGAL et sur des collaborations comme celles que nous avons menées avec David Roussel.

2 Analyse syntaxique

Un premier point d'approfondissement prometteur nous paraît être l'étude de l'intégration de notre méthode d'analyse par connexité dans une méthodologie du type de celle de [Billot et al.89] et [Lang91]. En pratique il s'agit d'étudier comment notre stratégie d'analyse s'exprime après une précompilation d'une grammaire TAG en LIG, dont les étapes de reconnaissance sont plus élémentaires. Une telle démarche permettrait d'offrir, parmi les différents algorithmes possibles, un algorithme plus adapté à une application robuste et qui nécessiterait des rattrapages particuliers, en plus d'algorithmes classiques de type CKY et Earley. Ceci pourrait par exemple s'exprimer dans le cadre de la démarche de [dlC et al.98] et [Alonso et al.99] fondée sur l'utilisation de SD-2SA (*Strongly Driven 2-Stack Automata*). Il serait également possible de comparer les résultats obtenus par rapport aux autres stratégies d'analyse avec plus de rigueur (nombre d'*items*, partages dans la forêt des dérivations).

En parallèle à un cadre indépendant de la stratégie, le formalisme des RCG proposé par [Boullier98], nous paraît être un formalisme très prometteur, notamment en tant que formalisme de compilation. Les grammaires hors contextes comme les TAG peuvent être compilées en RCG et être analysées selon leurs complexités au pire des cas habituels. La compilation de formalismes encore mal connus comme les TAG synchrones et les DTG permettrait d'obtenir des analyseurs efficaces et modulaires à partir d'un analyseur générique de RCG.

3 Gestion des ressources linguistiques

Une limite importante de notre travail concerne de façon générale les ressources mises en œuvre pour les LTAG :

- améliorer les accès aux différents lexiques, la solution la plus performante restant l'emploi de transducteurs à états finis ;
- réaliser des optimisations stochastiques de notre algorithme d'analyse afin d'en améliorer les performances moyennes.

Sur le dernier point, la nécessité de corpus en langue française annotés syntaxiquement du français afin d'optimiser les analyses est un besoin crucial. Il n'existe pas encore de vastes corpus annotés syntaxiquement, que ce soit en langue écrite ou orale, qui permettrait cette amélioration. Des travaux sont en cours à TALaNa en vue de développer ce style de corpus pour le français. Nous y prenons part en définissant des principes de codages incrémentaux et en développant des outils de visualisation et d'exploitation des annotations linguistiques.

L'annotation syntaxique de corpus est également importante pour tester la justesse des analyses et des réparations fournies. Un corpus de référence permet des calculs de précision et de correction qui sont des métriques indispensables pour une évaluation correcte de la grammaire et de l'analyseur.

4 Intégration dans un système de dialogue

Les formalismes de modélisation des différents niveaux d'analyse intervenant dans un système de dialogue, même complexes, peuvent faire l'objet de pré-compilations basées sur des modèles d'états finis. En faisant appel à des techniques complémentaires, ceci ne présuppose en rien une simplification des modèles et permet d'isoler des propriétés algorithmiques pouvant amener à des optimisations intéressantes. À partir de cette constatation, nous avons proposé un principe de synchronisation d'automates en vue d'intégrer l'analyseur syntaxique proposé avec les différents niveaux de traitement d'un système de dialogue oral. Ce besoin d'intégration est le plus souvent jugé essentiel, par exemple pour rendre compte des relations entre la syntaxe et la prosodie. En particulier, il nous paraît, très intéressant de pouvoir exploiter l'architecture proposée dans la partie 5 pour étudier la reconnaissance des phénomènes oraux à l'aide essentiellement d'une collaboration entre syntaxe et module de segmentation.

Nous pouvons ajouter que l'analyseur est en cours d'intégration dans un système d'interrogation d'annuaire d'entreprise intelligent nommé *MICoD*, développé par le département recherche et développement d'Alcatel Business Systems (ABS, Illkirch) en collaboration avec l'équipe Langue et Dialogue du LORIA (Nancy). Les ressources employées par cet analyseur sont également en cours de développement à partir de nos outils. Ceci confirme l'intérêt applicatif de nos travaux, tout en offrant un cadre de validation supplémentaire réaliste.

5 Traitement de l'écrit et de l'oral

Bien que les contraintes entre analyse de l'oral dans un contexte de dialogue et analyse de textes tout venant soient différentes, il nous semble particulièrement important de ne pas se dissocier de la communauté travaillant sur l'écrit. Les deux problématiques peuvent en effet être mutuellement profitables et ouvrir des perspectives conduisant à d'importants progrès : tout d'abord à l'aide des travaux sur l'écrit, l'analyse de l'oral gagnera en couverture et en finesse des interprétations linguistiques. Réciproquement, les travaux sur l'oral sont une source potentielle de solutions et de techniques robustes et efficaces pour l'écrit.

Annexes

Annexe A

Compléments sur les formalismes syntaxiques et les grammaires d'unification

A.1 Classification des grammaires formelles

A la suite des travaux de Chomsky, une classification des grammaires formelles a été établie en fonction de la forme des règles qu'elles mettent en œuvre. On peut rappeler rapidement qu'une grammaire formelle est définie par un quadruplet (T,N,x,P), où :

- T est un ensemble de symboles fini appelé vocabulaire terminal,
- N est un ensemble de symboles fini, non vide et disjoint de T appellé vocabulaire non-terminal,
- x est un élément particulier de N appelé axiome,
- P est une ensemble fini et non vide de règles de réécriture.

La classification est alors la suivante :

- les grammaires les plus générales, sans aucune contrainte imposée aux règles de réécriture, sont les grammaires de type 0, ou **grammaires semi-thuéiens**.
- les grammaires de type 1 ne contiennent que des règles de la forme :

  $$XqY \rightarrow XZY, \text{ avec } X, Y, Z \in (U \cup T)^* \text{ et } q \in N$$

  Ces grammaires sont appelées **grammaires contextuelles** (ou dépendantes du contexte).
- les grammaires de types 2 sont plus contraintes, puisqu'elles ne peuvent contenir que des règles de la forme :

  $$q \rightarrow X, \text{ avec } q \in N \text{ et } X \in (U \cup T)^*$$

  Ce sont les **grammaires hors contextes**.
- enfin, les grammaires les plus contraintes sont celle de type 3. Elles ne contiennent que des règles de la forme :

  $$q \rightarrow a, \text{ et}$$
  $$\text{soit } q \rightarrow ar \text{ (cas d'une grammaire linéaire droite)}$$
  $$\text{soit } q \rightarrow ra \text{ (cas d'une grammaire linéaire gauche)}$$
  $$\text{avec } a \in T \text{ et } q, r \in N$$

  Ces grammaires sont les **grammaires régulières**.

Notons que dans ce mémoire, nous n'avons fait référence qu'aux grammaires contextuelles et hors contexte.

A.2 Les temps anciens

A.2.1 Grammaires Transformationnelles

L'utilisation d'un langage formel tel que ceux définis par Chomsky en 1957 ne permettait pas le traitement du langage naturel. Les grammaires transformationnelles sont apparues en 1965 avec la théorie standard [Chomsky64]. Une transformation consiste en l'application d'une ou plusieurs transformations élémentaires (de type suppression, ajout, déplacement, ou substitution d'un élément) pour aboutir à une nouvelle structure syntagmatique.

L'objectif de Chomsky était de mettre en évidence deux niveaux de structures liées entre elles par transformations : une structure « profonde » et une structure « de surface ». Les similarités peuvent alors être décrites entre les structures profondes de phrases dont les structures de surface sont différentes. Cependant la surpuissance des grammaires transformationnelles (équivalente à des grammaires de type 0) les rend inefficaces pour des applications informatiques.

A.2.2 Grammaires de Cas

En linguistique traditionnelle, les recherches consacrées aux cas étaient essentiellement orientées vers l'étude des langues sur la base des systèmes de cas grec et latin. Fillmore, dans [Fillmore68], redéfinit la notion de cas, qu'il considère comme une relation entre des éléments intérieurs d'une phrase. Une phrase est avant tout constituée d'un verbe et d'un ou plusieurs groupes nominaux qui entretiennent des relations casuelles avec le verbe. Il remet en question les catégories traditionnelles comme le sujet ou le complément d'objet qu'il considère comme des réalisations de surface de fonctions plus profondes.

Ce modèle permet d'obtenir des structures prédicatives bien adaptées à une analyse sémantique, cependant il ne fournit aucune information concernant l'ordre des mots, et la définition des cas, difficile à généraliser, manque de critères de test objectifs.

A.2.3 Grammaires fonctionnelles

Les grammaires fonctionnelles[59], proposées notamment par S. Dik, se sont développées d'une manière relativement indépendante de la linguistique chomskyenne. Dans cette théorie, le langage est considéré comme une activité sociale liée à son contexte d'utilisation et le plus important est de tenir compte des aspects fonctionnels de la langue, c'est-à-dire déterminer le rôle des différents constituants en adéquation pragmatique et psychologique par rapport aux buts. Une grammaire fonctionnelle est constituée de trois types de fonctions : les fonctions sémantiques définissant les rôles des participants, les fonctions syntaxiques qui expriment comment peuvent être présentées les situations et les fonctions pragmatiques définissant le statut informationnel.

59. aussi désignés grammaires systémiques.

La génération de phrases se fait essentiellement en trois étapes :

1. Sélection d'un schéma de prédicat afin de produire une formule.
2. Transformation de la formule en formule entièrement spécifiée par affectation de fonctions syntaxiques et pragmatiques.
3. Application de règles d'expression à la formule afin de fournir comme résultat une phrase.

Par l'approche sémantique et pragmatique de la langue qu'elles permettent, les grammaires fonctionnelles constituent un modèle de la langue bien adapté pour représenter le sens d'un énoncé. Cependant, ce modèle est avant tout orienté vers la génération de phrases et non vers l'analyse et l'interprétation.

A.3 Notre temps : les grammaires d'unification

A.3.1 Introduction

Issus de développements historiques très divers, les formalismes d'unification sont apparus à partir du milieu des années 70. Rompant avec les modèles transformationnels de Chomsky, ces formalismes ont en commun l'usage d'un mécanisme particulier qui est l'*unification* de structures de traits.

L'objectif de cette partie, principalement inspirée de [Abeille93] et [Miller et al.90] est de présenter le cadre commun des grammaires d'unification. Nous nous intéresserons tout d'abord aux structures de traits manipulées qui peuvent être plus ou moins complexes, puis à l'opération d'unification proprement dite. Nous présenterons enfin un court historique des formalismes présentés dans ce rapport.

A.4 Principes de base

A.4.1 Structures de traits

Les Grammaire de Clauses Définies ou DCG et le formalisme des Grammaires Fonctionnelles d'Unification ou FUG, furent les deux premiers formalismes ayant introduit l'utilisation de structures de traits pour représenter les unités linguistiques manipulées en vue de mettre en œuvre l'opération d'unification. Ces structures ou matrices de traits sont un ensemble de couples attributs-valeurs notés entre crochets. Les attributs peuvent être à valeur atomique (c'est-à-dire valués avec un terme simple) ou complexe (c'est dire valués avec une structure de traits enchâssées). Par exemple, la description d'un groupe nominal, ou syntagme nominal, pourra prendre la forme décrite figure A.1 [Abeille93].

Une structure est mal formée si elle contient deux fois le même attribut au même niveau d'enchassement avec une valeur différente.

Une autre représentation des structures de traits couramment utilisée pour faciliter l'implantation informatique, est celle de **graphes acycliques orientés** : les noms des traits sont portés par les arcs et les nœuds portent la valeur des traits (voir figure A.2). Une

$$\begin{bmatrix} \text{Cat} = \text{GN} \\ \text{Nombre} = \text{x} \\ \text{Genre} = \text{y} \\ \text{det} = \begin{bmatrix} \text{Cat} = \text{D} \\ \text{Nombre} = \text{x} \\ \text{Genre} = \text{y} \end{bmatrix} \\ \text{Nom} = \begin{bmatrix} \text{Cat} = \text{N} \\ \text{Nombre} = \text{x} \\ \text{Genre} = \text{y} \end{bmatrix} \end{bmatrix}$$

FIG. A.1 – *Description d'un GN sous forme de structure de traits.*

structure est mal formée si deux arcs portant la même étiquette pointent à partir du même nœud sur des nœuds différents.

ce :

$$\begin{bmatrix} \text{Cat} = \text{D} \\ \text{Lex} = \text{'ce'} \\ \text{Accord} = \begin{bmatrix} \text{Genre} = \text{masc} \\ \text{Num} = \text{sing} \end{bmatrix} \end{bmatrix}$$

lapin :

$$\begin{bmatrix} \text{Cat} = \text{N} \\ \text{Lex} = \text{'lapin'} \\ \text{Accord} = \begin{bmatrix} \text{Genre} = \text{masc} \\ \text{Num} = \text{sing} \end{bmatrix} \end{bmatrix}$$

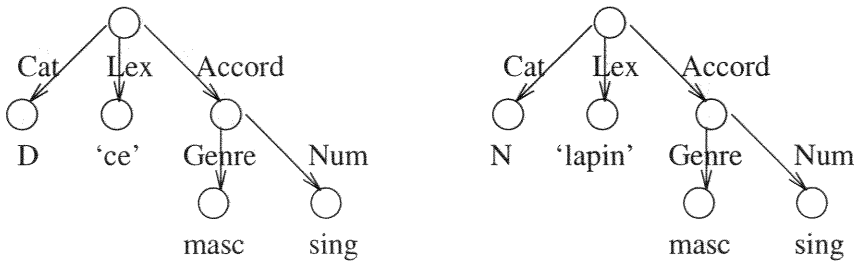


FIG. A.2 – *Structures de traits sous forme de matrices et de graphes associées à ce lapin.*

Lorsqu'on manipule des structures de traits complexes, il est possible que certains attributs partagent la même valeur. Les structures à valeurs partagées (ou *réentrantes*) sont coindexées comme on peut le voir figure A.3, A comporte deux attributs à valeur identique, alors que dans B les traits *Accord* partagent la même valeur et sont coindexés .

Le résultat de l'unification de A et B avec une structure C : [*Det* = [*Accord* = [*Genre* = *masc*]]], n'aboutira pas au même résultat. C unifiée avec A ne modifiera que la valeur du trait Det, alors que C unifiée avec B modifiera les deux structures de traits Det et Nom

$$\begin{array}{ll} \text{A :} & \text{B :} \\ \left[\begin{array}{l} \text{Det} = [\ \text{Accord} = [\ \text{Num} = \text{sing}\ ]\ ] \\ \text{Nom} = [\ \text{Accord} = [\ \text{Num} = \text{sing}\ ]\ ] \end{array}\right] & \left[\begin{array}{l} \text{Det} = [\ \text{Accord} = 1\ [\ \text{Num} = \text{sing}\ ]\ ] \\ \text{Nom} = [\ \text{Accord} = [1]\ ] \end{array}\right] \end{array}$$

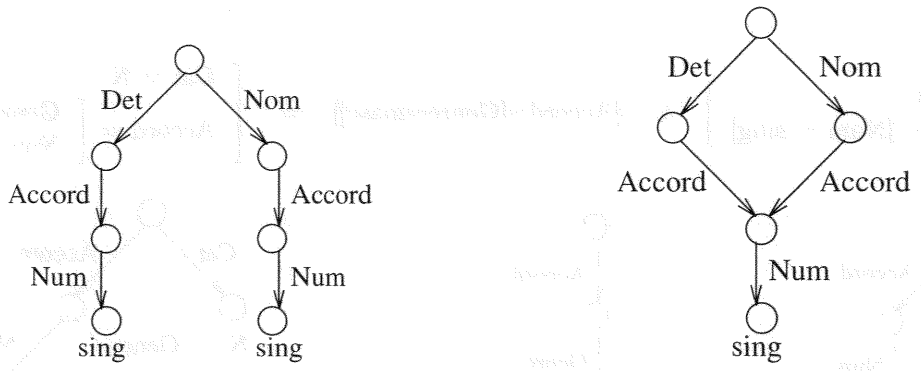


FIG. A.3 – *Représentation d'une structure à valeur partagée*

dont la valeur est partagée :

C unifiée avec A :

$$\left[\begin{array}{l} \text{Det} = \left[\begin{array}{l} \text{Accord} = \left[\begin{array}{l} \text{Num} = \text{sing} \\ \text{Genre} = \text{masc} \end{array}\right] \end{array}\right] \\ \text{Nom} = [\ \text{Accord} = [\ \text{Num} = \text{sing}\ ]\ ] \end{array}\right]$$

C unifiée avec B :

$$\left[\begin{array}{l} \text{Det} = \left[\begin{array}{l} \text{Accord} = 1\ \left[\begin{array}{l} \text{Num} = \text{sing} \\ \text{Genre} = \text{masc} \end{array}\right] \end{array}\right] \\ \text{Nom} = [\ \text{Accord} = [1]\ ] \end{array}\right]$$

A.4.2 Unification

On définit classiquement l'unification à l'aide de la notion d'**extension** entre structures de traits, une structure de traits A est une extension d'une structure de traits B, notée A $\supset$ B, ssi :

- tous les traits à valeur atomique présents dans B sont présents dans A avec la même valeur ;
- pour tout trait ayant une valeur non atomique, sa valeur dans A est une extension de sa valeur dans B.

Ainsi, si on considére la structure de trait associée à *lapin*, elle est une extension de la structure de traits à droite ci-dessous :

$$\left[\begin{array}{l} \text{Cat} = \text{N} \\ \text{Accord} = \left[\begin{array}{l} \text{Genre} = \text{masc} \\ \text{Num} = \text{sing} \end{array}\right] \end{array}\right] \supset \left[\begin{array}{l} \text{Cat} = \text{N} \\ \text{Accord} = [\ \text{Genre} = \text{masc}\ ] \end{array}\right]$$

L'**unification** de deux structures de traits A et B, noté A $\cup$ B, est la struture minimale qui est à la fois une extension de A et de B. Si une telle structure n'existe pas, l'unification

échoue. On peut donc comparer l'unification au calcul du plus petit commun multiple (PPCM) dans l'arithmétique. Pour illustrer cette définition, on peut donner les exemples ci-dessous :

$$[\text{Cat} = \text{D}] \cup [\text{Cat} = \text{N}] = \text{échec}$$

$$\begin{bmatrix} \text{Cat} = \text{N} \\ \text{Accord} = [\text{Num} = \text{sing}] \end{bmatrix} \cup [\text{Accord}=[\text{Genre}=\text{masc}]] = \begin{bmatrix} \text{Cat} = \text{N} \\ \text{Accord} = \begin{bmatrix} \text{Genre} = \text{masc} \\ \text{Num} = \text{sing} \end{bmatrix} \end{bmatrix}$$

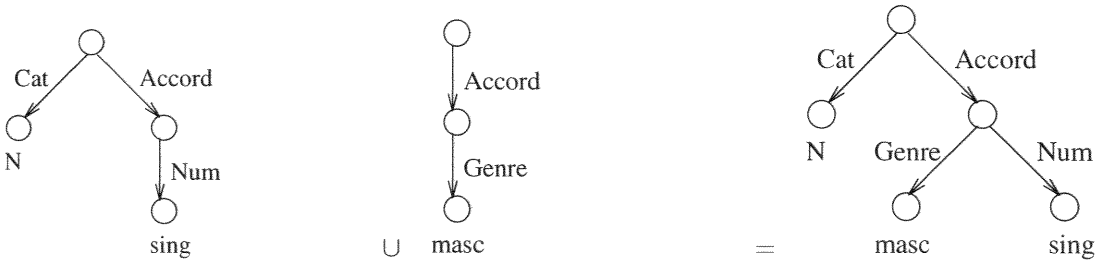


FIG. A.4 – *Exemples d'unification de structures de traits*

A.4.3 Complexité de l'unification

L'unification de termes finis est un problème décidable. Les algorithmes d'unification exponentiels par rapport à la longueur des termes sont les plus utilisés. Il existe également des algorithmes polynomiaux en temps et en espace. Ces derniers ne sont pas exploitables à cause de constantes trop importantes. Il existe même des algorithmes linéaires pour l'unification [Paterson et al.78], mais en pratique inutilisables. Dans le cas de termes infinis, les algorithmes existants sont d'une complexité exponentielle.

A.4.4 Origines et évolutions

A l'origine de la conception des grammaires d'unification, on trouve avant tout les travaux en programmation logique et une motivation linguistique qui est celle de pouvoir décrire la langue à l'aide du mécanisme simple et universel que constitue l'unification. Différents formalismes syntaxiques sont donc à l'origine des grammaires d'unification. Ils sont la conséquence de travaux menés en réaction contre les grammaires transformationnelles de l'école chomskyenne. La définition des grammaires de clauses définies (DCG) par F. Pereira et D. Warren en 1980, est issue des recherches en programmation logique qui ont notamment abouti à la mise au point du langage Prolog et de son mécanisme d'unification des termes. Les règles de réécriture y sont conçues comme des axiomes, et les phrases à analyser sont considérées comme des théorèmes. L'analyse syntaxique consiste à donner une preuve de ces théorèmes par inductions successives à partir des axiomes.

Le formalisme des grammaire fonctionnelles d'unification (FUG) a été introduite par [Kay80]. Il manipule des structures de traits identiques à celles présentées précédemment pouvant être à valeur non atomique. Il s'agit d'un outil linguistique non spécialisé opérant

uniquement par unification. Le résultat d'une analyse syntaxique à l'aide de ce formalisme n'est pas un classique arbre de dérivation, mais une structure de graphe. Dans le même esprit, Shieber proposa en 1981, le formalisme PATR, considéré comme la plus simple des grammaires d'unification. Le principal intérêt de PATR est d'offrir un ensemble d'outils d'analyse automatique basés sur les structures de traits et l'unification permettant de concevoir des grammaires de style différent.

La principale limite commune au trois formalismes DCG, FUG et PATR est qu'ils ne font pas appel à des principes linguistiques permettant de décrire de manière plus fine la langue. L'amélioration de la couverture de ces formalismes passe en effet par une multiplication des règles et des traits utilisés, entrainant obligatoirement une surgénération très importante d'hypothèses.

Nous allons sommairement évoquer des modèles de grammaires d'unification véritablement développées comme des théories linguistiques et non comme des outils de traitement du langage :

- les Grammaires Lexicales Fonctionnelles, LFG, 1982, issues des travaux de Bresnan qui prônait une linguistique non transformationnelle et fortement lexicaliste. Les LFG furent conçues en s'intéressant plus particulièrement aux représentations mentales des strutures linguistiques et aux contraintes universelles de la langue naturelle.
- les Grammaires Syntagmatiques Généralisées, GPSG, 1985, proposées par Gazdar. Il s'agit d'un formalisme basé sur des contraintes très générales, mais dont l'objectif était de traiter une grande variété de phńomènes jusqu'alors problématiques.
- les Grammaires Syntagmatiques guidées par les Têtes, HPSG, 1987, enrichit et systèmatise l'utilisation de structures de traits de GPSG.
- les Grammaires d'Arbres Adjoints, TAG, 1988, mettent en œuvre des unités linguistiques qui sont des arbres pouvant se combiner par adjonction ou substitution, opérations contraintes par l'unification.

Parallélement à ces travaux, d'autres formalismes d'origine plus ancienne se sont développés, enrichis par ces nouvelles idées mais en influançant également les grammaires d'unification. Ainsi, dans le formalisme des Grammaires Catégorielles d'Unification (UCG), on combine les techniques utilisées dans les grammaires basées sur l'unification, et celle des grammaires catégorielles comme le calcul de Lambek. Enfin, issu des grammaires transformationnelles, on trouve le modèle Gouvernement et Liage (GB), proposé par Chomsky en 1981, qui propose une grammaire modulaire basée sur des principes très généraux.

Annexe B

Approches probabilistes de l'analyse syntaxique

B.1 Introduction

Des méthodes probabilistes ont été employées très tôt dans le domaine du traitement automatique du langage naturel, cependant la puissance des ordinateurs a longtemps été un frein à l'efficacité de ces approches. L'étape d'apprentissage qui permet la construction de modèles en effet nécessite une quantité très importante de données.

Depuis le début des années 80, on dispose de machines suffisemment puissantes pour mettre en œuvre des analyses de très vastes corpus. Les modèles probabilistes ont permis d'apporter des solutions particulièrement efficaces en particulier dans le domaine de la reconnaissance de la parole (par exemple au niveau phonétique, où les modèles de Markov constituent la solution la plus performante).

Ce chapitre propose une présentation des principaux modèles d'analyse syntaxique probabiliste. Nous étudierons dans les deux parties suivantes le principe de ces modèles probabilistes, puis leur intérêt. Nous présentons ensuite les principaux modèles utilisés : modèles de Markov, modèles à base de n-grammes et les grammaires stochastiques.

B.2 Principe

Les mots d'une phrase suivent un ordre dépendant de contraintes syntaxico-sémantiques et ne sont donc pas combinés de n'importe quelle manière. Un modèle probabiliste est adapté pour prendre en compte ce style de problème, en l'occurrence le phénomène de succession des mots [Smaili91]. La probabilité de rencontrer un mot dépend de la première partie de la phrase en cours d'analyse.

Les méthodes probabilistes suivent toutes la même méthodologie que l'on peut exprimer en quatre étapes [Merialdo95] :

1. Identification du problème à résoudre : par exemple pour l'étiquetage grammatical, il s'agira de choisir la bonne classe d'un mot dans un contexte donné parmi les classes possibles.

2. Modélisation du problème : il s'agit de faire apparaître les probabilités de certains événements, en général à partir de probabilités élémentaires et de formules de calcul obtenues en faisant des hypothèses simplificatrices (par exemple, la probabilité de rencontrer un mot ne dépend que des deux mots précédents). Cette étape est particulièrement importante pour la qualité du modèle obtenu, et nécessite une bonne intuition du phénomène réel.
3. Détermination des valeurs des probabilités élémentaires par apprentissage.
4. Application du modèle à de nouvelles données.

B.3 Interêts

Deux aspects des méthodes probabilistes sont particulièrement intéressants pour le traitement de la langue : la résolution des ambiguïtés et l'apprentissage.

B.3.1 Résolution des ambiguïtés

Une des principales difficultés rencontrée dans le traitement automatique du langage naturel est, avec la question de la couverture linguistique, la résolution des ambiguités d'analyse. Ce problème a été présenté dans le chapitre 3 de ce rapport. Afin de prendre en compte un maximum de phénomènes syntaxiques, les formalismes proposés aboutissent le plus souvent une surgénération qui peut se révéler très importante. Le problème est alors de déterminer parmi toutes les structures syntaxiques obtenues celles qui correspondent à des constructions syntaxiques correctes et non liées à des ambiguïtés artificielles.

L'intérêt des méthodes probabilistes sur ce point est de fournir pour chaque solution potentielle une probabilité qu'il est possible d'interpréter comme une fréquence d'apparition de cette solution. En choisissant l'hypothèse ayant la plus forte probabilité, on minimise le risque d'erreur et on peut résoudre ainsi l'ambiguité.

B.3.2 Apprentissage

La capacité d'apprentissage représente un intérêt important des approches probabilistes. Grâce à l'amélioration constante des performances des ordinateurs, il est possible de construire des modèles probabilistes d'une qualité de plus en plus satisfaisante : d'une part on peut réaliser l'apprentissage avec une corpus de plus en plus vaste, et d'autre part on peut mettre au point des modèles de plus en plus complexes (utilisant plus de paramètres ou des structures comme des arbres).

B.4 Les modèles de Markov

Simples et efficaces, les modèles de Markov constituent la méthode probabiliste la plus utilisée. Avant de présenter les méthodes dérivées des modèles stochastiques markoviens appliqués au traitement de la parole, nous en rappellons dans cette partie inspirée de [Smaili91] le principe général.

Une chaîne de Markov est caractèrisée par un ensemble fini S d'états, dont deux états particuliers S_0 et S_f sont respectivement les états initial et final, un alphabet V et un ensemble T de transitions. Une transition (S_1,S_2,a) exprime le passage de l'état S_1 à l'état S_2 en produisant le symbole a. Deux matrices sont associées à une chaîne de Markov :

- une matrice A(n,n), où A(i,j) représente la probabilité de passer de l'état i à l'état j (transitions) ;
- une matrice B(n,m), où B(j,k) est la probalité d'observer le symbole k quand le processus est à l'état j.

La probabilité de suivre une suite de mots m_1, ..., m_n sera donnée par la formule :

$$P(m_1...m_n) = \sum_{i=1}^{p} \prod_{j=0}^{n} A(S_j^i,S_{j+1}^i) \text{ avec } S_{n+1}^i = S_f \text{ et } S_0^i = S_0 \forall i \in [1...p]$$

La difficulté de la conception d'une chaîne de Markov réside dans le choix du nombre d'états du modèle, en traitement du langage naturel ce nombre peut être très important. La phase d'apprentissage consiste à estimer les valeurs des probabilités des différentes transitions, c'est-à-dire déterminer les matrices A et B.

B.5 Modèle à base de n-grammes

La probabilité conditionnelle d'une phrase M composée de n mot M=$m_1...m_n$ est donnée par :

$$P(M) = \prod_{i=1}^{n} P(m_i|m_1...m_{i-1})$$

Il est en partique très difficile de mettre en œuvre cette probabilité dans un modèle probabiliste : la quantité d'information à fournir pour l'apprentissage nécessiterait un corpus d'une taille inestimable. C'est pourquoi on se restreint en général à la prise en compte des deux derniers mots précédant le mot en cours d'analyse (trigramme). La probabilité d'une phrase M sera alors :

$$P(M) = \prod_{i=2}^{n} P(m_i|m_{i-2}m_{i-1})$$

Cette réduction pose cependant toujours le problème de la taille du corpus d'apprentissage, c'est pourquoi on s'interessera en fait non pas à la probabilité de rencontrer un mot sachant les deux derniers mots analysés, mais sachant les deux dernières catégories grammaticales C_{i-2} et C_{i-1}. Les phrases sont alors des chaînes de Markov d'ordre 2.

B.6 L'étiquetage grammatical

L'étiquetage grammatical correspond au choix pour chaque mot de la catégorie grammaticale correcte dans le contexte où il apparaît. Plusieurs approches ont été proposées pour traiter ce problème : l'utilisation de modèles probabilistes, en général des trigrammes, a permis d'obtenir de bons résultats, avec des chiffres souvent supérieurs à 95% d'étiquetage correct.

Un système d'étiquetage est composé en général de trois éléments [Merialdo95] :

- un dictionnaire, qui permet de connaître pour chaque mot la liste des catégories grammaticales possibles dans le jeu de catégories considéré ;
- un modèle contextuel, soit sous forme de règles, soit sous forme de modèles probabilistes de type trigramme ou encore un réseau de neurones, permettant de prendre en compte le contexte pour l'étiquetage ;
- un algorithme enfin qui utilise ce modèle pour réaliser l'étiquetage.

Avec l'augmentation de la puissance de calcul des ordinateurs, les modèles probabilistes deviennent de plus en plus performants pour ce type d'application. Cependant, pour atteindre une performance optimale, il sera sans doute nécessaire d'intégrer des connaissances linguistiques suffisamment fines dans ce type de modèle.

B.7 Grammaires stochastiques

Les grammaires stochastiques sont des grammaires hors contextes dont les règles de réécriture $A_1 \rightarrow \chi_1$,...,$A_n \rightarrow \chi_n$ ont été munies de coeficients positifs p_1,p_2,...,p_n appellés probabilités des règles. Pour chaque catégorie non terminale A de G, on a :

$$\sum_{i/A_i=A} p_i = 1$$

Pour toute séquence M appartenant au langage reconnu par une grammaire stochastique G et tout arbre syntaxique T associé à M par G en appliquant les règles $R_1 ... R_m$, la probabilité p(T,M) est appellée *probabilité de dérivation* de M par T, et est par définition :

$$\mathrm{p(T,M)} = p_1 \cdots p_m$$

La probabilité d'une phrase M dans un tel modèle est alors définie comme la somme des probabilités de ses dérivations :

$$p(M) = \sum_{T \in T(M)} p(T,M)$$

où T(M) est l'ensemble des arbres possibles issus de l'analyse syntaxique de M.

Pour calculer la probabilité d'une phrase M, il suffit de construire toutes les dérivations de M possibles et de sommer leur probabilité. La probabilité de dérivation maximale parmi un ensemble possible correspond à la structure syntaxique la plus probable. On voit immédiatement l'intérêt d'un tel modèle pour gérer les problèmes d'ambiguïté [Rajman95].

Pour améliorer cette technique et ne pas avoir à calculer à chaque fois les probabilités de l'ensemble des structures syntaxiques possibles, on emploie en général des algorithmes classiques de l'analyse syntaxique (algorithme d'Earley, algorithme CKY,...) , mais modifiés de manière à factoriser, lors des calculs, les probabilités intermédiaires de parties commune à plusieurs différentes dérivations.

Afin de pouvoir construire le modèle stochastique, c'est-à-dire estimer les probabilités liées aux règles, il est nécessaire de disposer d'un corpus de phrases analysées, où pour chaque phrase un arbre syntaxique a été selectionné (généralement cette selection est manuelle).

L'intérêt des grammaires stochastiques par rapport aux trigrammes est une réduction importante du nombre de paramètres à évaluer, puisque qu'on n'a qu'une probabilité par règle. Le formalisme grammatical sous-jacent permet de prendre en compte des éléments syntaxiques moins localisés et plus complexes que dans le cas d'un modèle de Markov. Cependant certaines limitations sont à retenir concernant les grammaires stochastiques [Rajman95] : les grammairees hors-contextes sur lesquelles elles reposent peuvent vite amener à une augmentation du nombre de règles et de catégories à écrire. De plus, les grammaires stochastiques imposent une indépendance entre les règles de réécriture, les probabilités associées à l'application d'une règles sont indépendantes du contexte d'application de la règle. C'est pourquoi des modèles syntaxiques plus puissants comme les TAG peuvent s'avérer plus intéressants, de manière à prendre en compte les dépendances contextuelles.

B.8 Problèmes de mise en œuvre

La mise en œuvre de méthodes probabilistes se heurte à trois principales dificultés [Merialdo95] :

- la complexité des modèles est limitée par la puissance et la mémoire des machines, l'utilisation de statistique sur quatre mots est extrêmement difficile ;
- l'estimation des probabilités demande des corpus importants dont la conception bien souvent nécessite une intervention humaine. Ceci, compte-tenu de la taille des corpus permettant de bons résultats, est fort problématique ;
- enfin lorsque le nombre de paramètres d'un modèle est important (par exemple si on considère une suite de trois catégories parmi un jeu de 100 catégories, il y a un million de triplets envisageables, dont certainement plusieurs milliers peuvent se réaliser dans des textes), l'estimation des faibles propriétés est très imparfaite.

B.9 Conclusion

Pour tenter de résoudre les problèmes d'ambiguïtés lexicales, une solution est d'envisager une sophistication du modèle linguistique afin de permettre une meilleure représentation de la complexité des langages étudiés. Cependant la mise en œuvre de cette solution rend la tâche de modélisation particulièrement difficile, les délais s'évaluant alors en termes d'années. Une solution alternative est d'employer un modèle syntaxique probabiliste.

Les modèles à base de trigrammes ou de chaînes de Markov constituent un moyen simple pour mettre en œuvre des contraintes syntaxiques élémentaires, mais restent très limités. Les grammaires stochastiques permettent une description syntaxique plus complète, mais les résultats sont conditionnés par la qualité de l'apprentissage et du formalisme linguistique sous-jacent qui est obligatoirement une grammaire hors-contexte. L'introduction de formalismes plus puissants comme les LTAG permet la prise en compte contextuelle qui faisait défaut aux grammaires stochastiques, mais implique un nombre de paramètres à

estimer encore plus important. La possibilité d'utiliser un modèle probabiliste pour traiter les ambiguïtés syntaxiques et gagner en robustesse est une voie intéressante pour la robustesse de tout système de dialogue homme-machine.

Bibliographie

[Abeille et al.94] Abeillé (Anne), Daille (Béatrice) et Husson (A.). – FTAG : An implemented Tree Adjoining grammar for parsing French sentences. *TAG+3.* – Paris, 1994.

[Abeille et al.97] Abeillé (Anne) et Blache (Philippe). – Etat de l'art : la syntaxe. *Traitement Automatique des Langues (TAL)*, vol. 38, n° '27 2, 1997.

[Abeille et al.99] Abeillé (Anne) et Clément (Lionel). – A tagged reference corpus for french. *EACL workshop, Linguistically Interpreted Corpora (LINC-99), Bergen, Norway.* – 1999.

[Abeille91a] Abeillé (Anne). – Quand l'arbre ne cache pas la forêt : analyse du français à l'aide d'une grammaire d'arbres adjoints. *T. A. Informations*, vol. 32, n° '27 2, 1991, pp. 51–70.

[Abeille91b] Abeillé (Anne). – *Une grammaire lexicalisée d'arbres adjoints pour le français.* – Thèse de PhD, Université Paris 7, 1991.

[Abeille93] Abeillé (Anne). – *Les nouvelles syntaxes : grammaires d'unification et analyse du français.* – Armand Colin, Paris, 1993.

[Abney90] Abney (Steven). – Rapid incremental parsing with repair. *6th New OED Conference, University of Waterloo, Ontario.* – 1990.

[Abney91] Abney (Steven). – Parsing by chunks. – Kluwer Academic Publishers, 1991.

[Aho et al.72] Aho (A. V.) et Ullman (J.D.). – *The Theory of Parsing, Translation and Compiling.* – Prentice Hall, Englewood Cliffs, 1972.

[Alonso et al.99] Alonso (Miguel A.), Souto (David Cabrero), de la Clergerie (Eric) et Vilares (Manuel). – Tabular algorithms for tag parsing. *EACL'99, Bergen.* – 1999.

[AM et al.97] Aït-Mokhtar (Salah) et Chanod (Jean-Pierre). – Incremental finite-state parsing. *ANLP'97, Washington*, 1997.

[Antoine et al.99] Antoine (Jean-Yves) et Genthial (Damien). – Méthodes hybrides taln/talp. *Atelier TALN sur les techniques hybrides pour le traitement robuste de la langue.* – 1999.

[Antoine96] Antoine (Jean-Yves). – *Coopération syntaxe-sémantique pour la compréhension automatique de la parole spontanée.* – Thèse de PhD, Université INPG Grenoble, 1996.

[Baschung91] Baschung (Karine). – Un analyseur UCG du français. *T. A. Informations*, vol. 2-1991, 1991, pp. 71–88.

[BB90] Blanche-Benveniste (Claire). – *Le français parlé : études grammaticales.* – CNRS Éditions, 1990.

[Bear et al.92] Bear (J.), Dowding (J.) et Shriberg (E.). – Integrating multiple knowledge sources for detecting and correcting of repairs in human-computer dialog. *ACL*. – 1992.

[Becker91] Becker (Tilman). – Long-distance scrambling and tree adjoining grammars. *EACL'95, Berlin*. – 1991.

[Becker93] Becker (Tilman). – *HyTAG : A new Type of Tree Adjoining Grammars for Hybri Syntactic Representation of Free Word Order Languages*. – Thèse de PhD, University of Saarbrücken, 1993.

[Bellenger79] Bellenger (Lionel). – L'expression orale. *L'expression orale*. – Collection que sais-je, Presse Universitaire de France, 1979.

[Biber86] Biber (Douglas). – Spoken and written textual dimensions in english : resolving the contradictory findings. *Langage*, vol. 62, n˚ '27 2, 1986, pp. 384–414.

[Bilange92] Bilange (Eric). – *Dialogue personne-machine : modélisation et réalisation informatique*. – Hermès, Paris, 1992.

[Billot et al.89] Billot (Sylvie) et Lang (Bernard). – The structure of shared forests in ambigous parsing. *33rd Conference of the Association for Computional Linguistics (ACL'89)*. – 1989.

[Bod95] Bod (Rens). – *Enriching Linguistics with Statistics : Performance Models of Natural Language*. – Thèse de PhD, University of Amsterdam, 1995.

[Bonhomme et al.99] Bonhomme (Patrice) et Lopez (Patrice). – TagML : codage XML et ressources pour les Grammaires d'Arbres Lexicalisées. *draft*. – 1999.

[Boros et al.96] Boros (M.), Eckert (W.), Gallwitz (Florian), Görz (G.), Hanrieder (G.) et Niemann (Heinrich). – Toward understanding spontaneous speech : Word accuracy vs. concept accuracy. *ICSLP'96, Philadelphia*. – 1996.

[Boufaden98] Boufaden (Narjes). – *Analyse syntaxique robuste des textes de dialogues oraux*. – Thèse de PhD, Université Laval, 1998.

[Boullier96] Boullier (Pierre). – Another facet of lig parsing. *ACL'96, Santa Cruz*. – 1996.

[Boullier98] Boullier (Pierre). – A generalization of midly context-sensitive formalisms. *International Workshop on Tree Adjoing Grammars and related framework (TAG+4), Philadelphia*. – 1998.

[Boullier99] Boullier (Pierre). – On multicomponent tag parsing. *TALN'99, Cargèse, Corse*. – 1999.

[Briffault et al.97] Briffault (Xavier), Chibout (Karim), Sabah (Gérard) et Vapillom (Jérôme). – An Object-Oriented Linguistic Engineering Environment using LFG and Conceptual Graphs. *International Workshop ENVGRAM 97*. – Madrid, Spain, 1997.

[Briscoe et al.87] Briscoe (Edward), Grover (Claire), Branimir (Boguraev) et Carroll (John). – A formalism and environment for the development of a large grammar of english. *10th International Joint Conference on Artificial Intelligence, Milan, Italy*, pp. 703–708. – 1987.

[Brison et al.95] Brison (Éric) et Vigouroux (Nadine). – Robust comprehension in a spoken dialog system. *Eurospeech'95, Madrid*. – 1995.

[Bruneseaux et al.97] Bruneseaux (Florence) et Romary (Laurent). – Codage des référents et

coréférents dans les dialogues homme-machine. *Colloque ACH-ALLC'97, Kingston*, pp. 15–18. – 1997.

[cAM97] cois Arnould. Mathieu (Fran). – *Prise en compte de contraintes pragmatiques pour guider un système de reconnaissance de la parole : le système COMPPA.* – Thèse de PhD, Université Henri Poincaré Nancy 1, 1997.

[Candito et al.98] Candito (Marie-Hélène) et Kahane (Sylvain). – Defining dtg derivations to get semantic graphs. *Fourth International Workshop on TAG+ (TAG+4), Philadelphia.* – 1998.

[Candito96] Candito (Marie-Hélène). – A principle-based hierarchical representation of LTAGs. *COLING'96.* – Copenhagen, Denmark, 1996.

[Candito99] Candito (Marie-Hélène). – *Structuration d'une grammaire LTAG : application au français et à l'italien.* – Thèse de PhD, University of Paris 7, 1999.

[Carberry89] Carberry (Sandra). – A Pragmatics-Based Approach To Ellipsis Resolution. *Computational Linguistics*, vol. 15, n '27 2, 1989, pp. 75–96.

[Carbonell et al.83] Carbonell (J.G.) et Hayes (P.J.). – Recovery strategies for parsing extra-grammatical language. *Americal Journal of Computational Linguistics*, vol. 9, 1983, pp. 123–146.

[Carbonell83] Carbonell (Jaime G.). – Discourse Pragmatics and Ellipsis Resolution in Task-oriented Natural Languages Interfaces. *ACL'83.* – Cambridge, 1983.

[Carre et al.91] Carré (R.), Dégremont (J.-F.), Gross (M.), Pierrel (J.-M.) et Sabah (G.). – *Langage humain et machine.* – Presses du CNRS, Paris, 1991.

[Carroll et al.99a] Carroll (John), Nicolov (Nicolas), Shaumyan (Olga) et Smets (Martine). – Parsing with an extended domain of locality. *Eighth Conference of the European chapter of the Association for Computational Linguistics (EACL'99), Bergen, Norway.* – 1999.

[Carroll et al.99b] Carroll (John), Nicolov (Nicolas), Shaumyan (Olga), Smets (Martine) et Weir (David). – Efficient parisng with wide-coverage lexicalized grammars. *Submitted to ACL'99.* – 1999.

[Carroll83] Carroll (John). – An island parsing interpreter for full augmented transition network formalism. *First Conference of the European Chapter of The Association for Computational Linguistics, Pisa, Italy*, pp. 101–105. – 1983.

[Carroll93] Carroll (John). – *Practical Unification-based Parsing of Natural Language.* – Thèse de PhD, University of Cambridge, 1993.

[Chafe85] Chafe (Wallace L.). – Linguistic differences produced by differences between speaking and writing. *Literacy, language and learning : The nature and consequences of reading and writing*, éd. par Olson (D.R.), Torrance (N.) et Hildyard (A.), pp. 105–123. – Cambridge University Press, 1985.

[Chanod93] Chanod (Jean-Pierre). – Problèmes de robustesse en analyse syntaxique. *Conférence Informatique et Langue Naturelle*, pp. 223–244. – 1993.

[Chapelier96] Chapelier (Laurent). – *Dialogue d'assistance dans une interface homme-machine multimodale.* – Thèse de PhD, Université Henri Poincaré Nancy I, 1996.

[Chen et al.99] Chen (John), Bangalore (Srinivas) et Vijay-Shanker (K.). – New models for improving supertag disambiguation. *EACL'99, Bergen, Norway.* – 1999.

[Chomsky64] Chomsky (Noam). – *Aspects de la théorie syntaxique.* – Seuil, 1964.

[Ciravegna et al.97] Ciravegna (Fabio), Lavelli (Alberto), Petrelli (Daniela) et Pianesi (Fabio). – Participatory Design for Linguistic Engineering : the case of the Geppetto Development Environment. *Workshop ENVGRAM.* – Madrid, Spain, 1997.

[Clemenceau93] Clémenceau (David). – *Structuration du lexique et reconnaissance de mots dérivés.* – Thèse de PhD, Université Paris 7, 1993.

[Cohen et al.94] Cohen (J.), Gish (H.) et Flanagan (J.). – Switchboard, the second year. *CAIP summer workshop in speech recognition : Frontiers in Speech Processing II.* – 1994.

[Colineau97] Colineau (Nathalie). – *Étude des marqueurs discursifs dans le dialogue finalisé.* – Thèse de PhD, Université Joseph Fourier, Grenoble, 1997.

[Cori et al.97] Cori (Marcel), de Fornel (Michel) et Marandin (Jean-Marie). – Parsing Repairs. *Recent advances in natural language processing*, éd. par Mitkov (Ruslan) et Nicolov (Nicolas). – John Benjamins, 1997.

[Danlos98] Danlos (Laurence). – GTAG : un formalisme lexicalisé pour la génération de textes inspiré de TAG. *Traitement Automatique des Langues - TAL*, vol. 39, n° '27 2, 1998.

[Dessales93] Dessales (Jean-Louis). – *Modèle cognitif de la communication spontanée appliqué à l'apprentissage des concepts.* – Thèse de PhD, ENST, Paris, 1993.

[Deville89] Deville (Guy). – *Modelization of task-Oriented Utterances in a Man-Machine Dialogue System.* – Thèse de PhD, University of Antwerpen, Belgique, 1989.

[dlC et al.98] de la Clergerie (Eric) et Pardo (Miguel Alonso). – A tabular interpretation of a 2-stack automata. *Coling/ACL'98, Montréal.* – 1998.

[Doran et al.94] Doran (Christy), Egedi (Dania), Hockey (Beth Ann), Srinivas (B.) et Zaidel (Martin). – XTAG System - A Wide Coverage Grammar for English. *COLING.* – Kyoto, Japan, 1994.

[Earley70] Earley (Jay C.). – An efficient context-free parsing algorithm. *Communication of the ACM*, vol. 13, n° '27 2, 1970, pp. 94–102.

[Evans et al.95] Evans (Roger), Gazdar (Gerald) et Weir (David). – Encoding lexicalized tree adjoining grammar with a nonmonotonic inheritance hierarchy. *ACL'95.* – 1995.

[Evans et al.97] Evans (Roger) et Weir (David). – Automaton-based Parsing for Lexicalized Grammars. *Fifth International Workshop on Parsing Technologies.* – Cambridge, Mass, 1997.

[Evans et al.98] Evans (Roger) et Weir (David). – A structure-sharing parser for lexicalized grammars. *COLING-ALC.* – Montréal, Canada, 1998.

[Falzon84] Falzon (P.). – Les langages opératifs. *séminaire GRECO sur le dialogue oral homme-machine, Nancy.* – 1984.

[Fillmore68] Fillmore (C. J.). – *The Case for Case.* – Universals in Linguistic Theory, New York, 1968.

[Freud01] Freud (Sigmund). – *The Psychopathology of Everyday Life.* – 1901.

[Frey89] Frey (Corina). – L'anaphore et l'ellipse. pp. 169–213. – Publications de la Sorbonne nouvelle, 1989.

[Fuchs et al.93] Fuchs (Catherine), Danlos (Laurence), Lacheret-Dujour (Anne), Luzzati (Daniel) et Victorri (Bernard). – *Linguistique et Traitements Automatiques des Langues.* – Hachette, 1993.

[Fung et al.75] Fung et Fu. – Stochastic syntactic decoding for pattern classification. *IEEE Trans. Comp.*, vol. 24, n° '27 6, 1975, pp. 662–669.

[Gadet89] Gadet (F.). – *Le Français ordinaire.* – Armand Colin, Paris, 1989.

[Gaiffe92] Gaiffe (Bertrand). – *Référence et dialogue homme-machine : vers un modèle adapté au multi-modal.* – Thèse de PhD, Université de Nancy 1, 1992.

[Gardent et al.93] Gardent (Claire) et Baschung (Karine). – *Techniques d'analyse et de génération pour la langue naturelle.* – ADOSA, 1993.

[Gazdar et al.85] Gazdar (Gerald), Klein (E.), Pollum (G.) et Sag (Ian). – *Generalized Phrase Structure Grammar.* – Harvard University Press, 1985.

[Gee et al.83] Gee (James) et cois Grosjean (Fran). – Performance structures : psycholinguistic and linguistic appraisal. *Cognitive Psycholog*, vol. 15, 1983, pp. 411–458.

[Goodman98] Goodman (Joshua). – *Parsing Inside-Outside.* – Thèse de PhD, Division of Emgineering and Applied Sciences, Harvard University, May, 1998.

[Gross75] Gross (Maurice). – *Méthode en syntaxe.* – Hermann, 1975.

[Halber98a] Halber (Ariane). – Grammatical factor and spoken sentence recognition. *Workshop on Text, Speech and Dialog, Brno.* – 1998.

[Halber98b] Halber (Ariane). – Tree Grammars Linear Typing for unified Super-Tagging/Probabilistic Parsing Model. *Workshop on Tree Adjoining Grammars (TAG+4).* – Philadelphia, USA, 1998.

[Halber99] Halber (Ariane). – *Unification de traits pour les FB-LTAG : synchronie DAG/TAG.* – Rapport technique, ENST (draft), juin 1999.

[Harbusch94] Harbusch (Karin). – Incremental natural-language processing with schema-tree adjoining grammars. *TAG+3, Paris.* – 1994.

[Harris51] Harris (Zellig S.). – *Methods in Structural Linguistics.* – University of Chicago Press, 1951.

[Heeman et al.94] Heeman (P.) et Allen (J.). – Detecting and correcting speech repairs. *ACL.* – 1994.

[Heeman et al.96] Heeman (P.), Loken-Kim (K.) et Allen (J.). – Combining the detection and correction of speech repairs. *ICSLP.* – 1996.

[Hepple99] Hepple (Mark). – A functional interpretation scheme for d-tree grammars. *Third International Workshop on Computational Semantics (IWCS-3), Tilburg.* – 1999.

[Hermansky90] Hermansky (H.). – Perceptual linear predictive (PLP) analysis of speech. *Journal of the Acoustical Society of America*, vol. 87, 1990, pp. 1738 – 1752.

[Hindle83] Hindle (D.). – Deterministic parsing of syntactic non-fluencies. *21th Annual Meeting of the Association of Computational Linguistics, Massachusetts.* – 1983.

[Hindle89] Hindle (D.). – Acquiring disambiguation rules from text. *27th Meeting of the Association for Computational Linguistics, Vancouver, Canada*, pp. 118–125. – 1989.

[Husson et al.96] Husson (Jean-Luc) et Laprie (Yves). – A new search algorithm in segmentation lattices of speech signals. *ICSLP'96, Philadelphia, USA*, 1996.

[Husson98] Husson (Jean-Luc). – *Une approche hiérarchique de la segmentation du signal de parole.* – Thèse de PhD, Université Henri Poincaré, Nancy, 1998.

[Husson99] Husson (Jean-Luc). – Evaluation of a segmentation framework based on multi-level lattices. *to appear in EUROSPEECH, Budapest, Hungary.* – 1999.

[Issac97] Issac (Fabrice). – *Analyse syntaxique et apprentissage des langues.* – Thèse de PhD, Université Paris-Nord, 1997.

[Issac98] Issac (Fabrice). – A Standard Representation Framework for TAG. *Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+4).* – 1998.

[Jelinek et al.94] Jelinek (Fred), Lafferty (John), Magerman (David M.), Mercer (Robert), Ratnaparkhi (Adwait) et Roukos (Salim). – Decision Tree Parsing using a Hidden Derivation Model. *ARPA Workshop on Human Langage Technology.* – 1994.

[Jensen et al.93] Jensen (Karen), Heidorn (George E.) et Richardson (Stephen D.). – *Natural Language Processing : The PLNLP Approach.* – Kluwer Academic Publishers, 1993.

[Joshi et al.75] Joshi (A.), Levi (L.) et Takahashi (M.). – Tree adjunct grammars. *Journal of the Computer and System Sciences*, 1975.

[Joshi et al.91] Joshi (Aravind K.), Vijay-Shanker (K.) et Weir (David). – The convergence of midly context-sensitive grammatical formalisms. *Foundational Issues in Natural Language Processing.* – MIT Press, 1991.

[Kahlil97] Kahlil (Sima'an). – Computational complexity of probabilistic disambiguations by means of tree-grammars. *ACL/EACL, Madrid.* – 1997.

[Kaplan et al.82] Kaplan (Ronald) et Bresnan (Joan). – *LFG : a formal system for grammatical representation.* – 1982.

[Kasper et al.95] Kasper (Robert), Kiefer (Bernd), Netter (Klaus) et Vijay-Shanker (K.). – Compilation of hpsg to tag. *ACL, Cambridge, Massachusetts.* – 1995.

[Kay80] Kay (Martin). – Algorithm schemata and data structure in syntactic processing. *Readings in Natural Language Processing, Morgan Kaufmann*, 1980, pp. 35–70.

[Kay86] Kay (Martin). – Algorithm schemata and data structures in syntactic processing. *Reading in Natural Language Processing.* – Morgan Kaufmann Publishers, Los Altos, California, 1986.

[Kinyon97] Kinyon (Alexandra). – Un algorithme d'analyse lr(0) pour les grammaires d'arbres adjoints lexicalisées. *TALN'97, Grenoble, France.* – 1997.

[KO96] Kerbrat-Orecchioni (Catherine). – *La conversation.* – Seuil, 1996.

[Kroch et al.85] Kroch (A.) et Joshi (A.). – *The linguistic relevance of Tree Adjoining Grammars.* – Rapport technique, University of Pennsylvania, Philadelphia, 1985.

[Lambrecht96] Lambrecht (K.). – On the formal and functional relationship between topics and vocatives. *Conceptual Structures, Discourse and Language*, pp. 267–288. – CSLI Publications, Stanford, 1996.

[Lamel98] Lamel (Lori). – Spoken language dialog system development and evaluation at limsi. *ISSD'98, Sydney.* – 1998.

[Lang74] Lang (Bernard). – Deterministic techniques for efficient non-deterministic parsers. *2nd Colloquim on Automata, Languages and Programming, Saarbrücken*, éd. par Loeckx (J.), pp. 255–269. – 1974.

[Lang88] Lang (Bernard). – Parsing incomplete sentences. *12th International Conference on Computational Linguistics (COLING'88)*, pp. 365–371. – 1988.

[Lang89] Lang (Bernard). – A generative view of ill-formed input processing. *ATR Symposium on Basic Research for Telephone Interpretation (ASTI), Kyoyo, Japon.* – 1989.

[Lang90] Lang (Bernard). – The systematic construction of earley parsers : Application to the production of $o(n^6)$ earley parsers for tree adjoining grammars. *First International Workshop on Tree Adjoining Grammars, Dagstuhl Castle, Germany.* – 1990.

[Lang91] Lang (Bernard). – Towards a Uniform Formal Framework for Parsing. *Current Issues in Parsing Technology*, éd. par Tomita (M.). – Kluwer Academic Publishers, 1991.

[Lang94] Lang (Bernard). – Recognition can be harder than parsing. *Computational Intelligence*, 1994.

[Lari et al.90] Lari (K.) et Young (S. J.). – The estimation of stochastic context-free grammars using inside-outside algorithm. *Computer Speech and Language Processing*, vol. 4, 1990, pp. 35–56.

[Lavelli et al.91] Lavelli (Alberto) et Satta (Giorgio). – Bidirectional parsing of lexicalized tree adjoining grammars. *Fifth Conference of the European Chapter of the Association for Computational Linguistics.* – Berlin, Germany, 1991.

[Lavie96] Lavie (Alon). – *GLR*: A Robust Grammar-Focused Parser for Spontaneously Spoken Language.* – Pittsburgh, Thèse de PhD, Carnegie Mellon University, 1996.

[Levelt89] Levelt (W.J.M.). – *Speaking.* – The MIT Press, 1989.

[Lopez et al.98] Lopez (Patrice) et Roussel (David). – Which rules for robust parsing of spoken utterances with Lexicalized Tree Grammars. *Workshop on Tree Adjoining Grammars (TAG+4).* – Philadelphia, USA, 1998.

[Lopez96] Lopez (Patrice). – *La syntaxe dans les interfaces intelligentes composante orale.* – Rapport de dea, UHP - Nancy 1, 1996.

[Lopez98a] Lopez (Patrice). – A LTAG grammar for parsing incomplete and oral utterances. *European Conference on Artificial Intelligence (ECAI).* – Brighton, UK, 1998.

[Lopez98b] Lopez (Patrice). – Analyse guidée par la connexité de TAG lexicalisées. *Conférence sur le Traitement Automatique du Langage Naturel (TALN'98).* – Paris, France, 1998.

[Lopez98c] Lopez (Patrice). – Connection driven parsing of Lexicalized TAG. *Workshop on Text, Speech and Dialog (TSD).* – Brno, Czech Republic, 1998.

[Luhan68] Luhan (Marshall Mac). – *Pour comprendre les médias.* – Seuil, Paris, 1968.

[Luzzati83] Luzzati (D.). – *Recherches sur la structure du discours oral spontané.* – Thèse de PhD, Université Paris III, 1983.

[Luzzati89] Luzzati (Daniel). – *Recherche sur le dialogue homme-machine : modèles linguistiques et traitements automatiques.* – Thèse de PhD, Université de la Sorbonne nouvelle Paris 3, 1989.

[Luzzati95] Luzzati (Daniel). – *Le dialogue verbal homme-machine : étude de cas.* – Sciences Cognitives, Masson, 1995, 267–288p.

[Magerman et al.92] Magerman (David M.) et Weir (Carl). – Efficiency, Robustness, and Accuracy in Picky Chart Parsing. *ACL'92.* – 1992.

[Marcus80] Marcus (M.). – *A theory of syntactic recognition for natural language.* – MIT Press, Cambridge, MA, 1980.

[Mela92] Mela (Augusta). – *Traitement automatique de la coordination par et.* – Thèse de PhD, Université de Paris Nord, 1992.

[Melcuk88] Mel'cuk (Igor). – *Dependency Syntax: Theory and Practice.* – Albny State University of New-York Press, 1988.

[Merialdo95] Merialdo (Bernard). – Modèles probabilistes et étiquetage automatique. *T.A. Informations*, vol. 36, 1995, pp. 7–22.

[Miller et al.51] Miller (G.A.), Heise (G.) et Lichten (W.). – The intelligibility of speech as a function of the context of the test materials. *Journal. exper. Psychol.*, vol. 41, 1951, pp. 329–335.

[Miller et al.90] Miller (Philip) et Torris (Thérèse). – *Formalismes syntaxiques pour le traitement automatique du langage naturel.* – Hermès, 1990.

[Minker96] Minker (Wolfgang). – Compréhension dans le domaine atis. *21ème Journées d'Études sur la Parole, JEP'96, Avignon, France*, pp. 417–420. – 1996.

[Minsky85] Minsky (Marvin). – *The Society of Mind.* – Touchstone Books, New York, 1985.

[Morel88] Morel (Marie-Annick). – *Anaphores et Ellipses. Analyse linguistique d'un corpus de dialogues Homme/Machine, Premier corpus : Centre de renseignements SNCF à Paris.* – Publication de la Sorbonne Nouvelle, 1988.

[Morel89] Morel (Marie-Annick). – *Analyse linguistique d'un corpus de dialogues Homme/Machine, Deuxième corpus : CIO Centre d'Information et d'Orientation de l'Université Paris V.* – Publications de la Sorbonne Nouvelle, 1989.

[Muller87] Muller (J. P.). – *Contribution l'étude d'un agent rationnel : spécification en logique intensionnelle et implantation.* – Thèse de PhD, Grenoble, 1987.

[Nakatani et al.94] Nakatani (C. H.) et Hirschberg (J.). – A corpus-based study of repair cues in spontaneous speech. *Journal of the Acoustical Society of America*, vol. 95, n° '27 3, 1994.

[Nederhof97] Nederhof (Mark-Jan). – Solving the correct-prefix property for ltags. *Fifth Meeting on Mathematics of Language (MOL5), Schloss Dagstuhl, Germany.* – 1997.

[Nederhof98] Nederhof (M.J.). – Indexed Automata and Tabulation of TAG Parsing. *Workshop on Tabulation in Parsing and Deduction.* – Paris, 1998.

[Newell et al.58] Newell (A.), Shaw (J. C.) et Simon (H. A.). – Elements of a theory of human problem solving. *Psychological Review*, vol. 65, 1958, pp. 151–166.

[Oshaughnessy92] O'Shaughnessy (D.). – Recognition of hesitations in spontaneous speech. *IEEE Conference on Acoustics, Speech and Signal Processing.* – 1992.

[Pallett et al.95] Pallett (D. S.), Fiscus (J. G.), Fisher (W.), Garofolo (J. S.), Lund (A.), Martin (A.) et Przybocki (M. A.). – 1994 benchmark tests for the arpa spoken language program. *ARPA workshop.* – 1995.

[Paterson et al.78] Paterson (M. S.) et Wegman (M. N.). – Linear unification. *Journal of Computer and Sciences*, vol. 16, 1978, pp. 158–167.

[Pereira et al.80] Pereira (Fernando) et Warren (David). – Definite clause grammars for language analysis - a survey of the formalism and a comparison with augmented transition networks. *Artificial Intelligence*, vol. 13, n° '27 3, 1980, pp. 231–278.

[Pereira et al.91] Pereira (Fernando C. N.) et Riley (Michael D.). – Speech recognition by composition of weighted finite automata. *Proceedings of ACL'91, Berkeley, USA.* – 1991.

[Pereira85] Pereira (Fernando). – A structure-sharing representation for unification-based grammar formalisms. *23rd Meeting of the Association for Computational Linguistics.* – 1985.

[Pierrel81] Pierrel (Jean-Marie). – *Étude et mise en œuvre de contraintes linguistiques en compréhension automatique du discours continu.* – Thèse de PhD, Université Nancy 1, 1981.

[Pierrel97] Pierrel (Jean-Marie). – Bilan des grandes orientations de recherche en traitement automatique du langage parlé en france. *Traitement Automatique des Langues (T.A.L.)*, vol. 38, n° '27 2, 1997, pp. 7–46.

[Pollard et al.94] Pollard (Carl) et Sag (Ivan). – Head-driven phrase structure grammar. *CSLI series.* – University of Chicago Press, 1994.

[Price et al.89] Price (Patti), Moore (Robert), Murveit (Hy), Pereira (Fernando), Bernstein (Jared) et Dalrymple (Mary). – The integration of speech and natural language in interactive spoken language systems. *Proceeding of Eurospeech.* – Paris, France, 1989.

[Rajman95] Rajman (Martin). – Approche probabiliste de l'analyse syntaxique. *T.A. Informations*, vol. 36, 1995, pp. 157–199.

[Rambow et al.95a] Rambow (Owen), Shanker (K. Vijay) et Weir (David). – D-tree grammars. *33rd Conference of the Association of Computational Linguistics (ACL'95)*, pp. 151–158. – 1995.

[Rambow et al.95b] Rambow (Owen), Shanker (K. Vijay) et Weir (David). – Parsing d-tree grammars. *Fourth International Workshop on Parsing Technologies (IWPT'95).* – 1995.

[Rambow et al.96] Rambow (Owen) et Satta (Giorgio). – Synchronous models of language. *ACL'96 Santa Cruz.* – 1996.

[Rambow94] Rambow (Owen). – *Formal and Computational Aspects of Natural Language Syntax.* – Thèse de PhD, University of Pennsylvania, 1994.

[Rastier91] Rastier (Francois). – *Sémantique et recherches cognitives.* – PUF, Paris, 1991.

[Reboul et al.98] Reboul (Anne) et Moeschler (Jacques). – *La pragmatique aujourd'hui. Une nouvelle science de la communication.* – le Seuil, 1998.

[Redeker84] Redeker (G.). – On the difference between oral and written communication. *Discourse Processes*, vol. 7, 1984, pp. 43–55.

[Resnik92] Resnik (Philip). – Probabilistic tree-adjoining grammars as a framework for statistic natural language processing. *COLING'92, Nantes, France*, 1992.

[rg98] research group (XTAG). – *A Lexicalized Tree Adjoining Grammar for English.* – Rapport technique, IRCS 98-03, University of Pennsylvania, 1998.

[Rice98] Rice (Jeff). – Literacy and orality are still in our times. *FCEA Conference.* – 1998.

[Rickley94] Rickley (R. J.). – *Detecting Disfluency in Spontaneous Speech.* – Thèse de PhD, University of Edinburgh, Scotland, 1994.

[Roche96] Roche (Emmanuel). – *Parsing with Finite-State Transducers.* – Rapport technique, TR-96-30, Mitsubishi Electric Research Laboratories (MERL), 1996.

[Rose et al.97] Rosé (C.P.) et Lavie (A.). – An efficient distribution of Labor in Two Stage Robust Interpretation Process. *Proceeding of Empirical Methods in Natural Language Processing, EMNLP'97.* – Rhode Island, USA, 1997.

[Roussel et al.97] Roussel (David) et Halber (Ariane). – Filtering errors and repairing Linguistic Anomalies for Spoken Dialogue Systems. *Workshop on Interactive Spoken Dialog Systems : ACL/EACL*, pp. 74–81. – Madrid, 1997.

[Roussel et al.98a] Roussel (David) et Modave (Francois). – A multicriteria scoring method to parse recognition hypotheses. *the International Workshop on Speech and Computer (SPECOM).* – St.-Petersburg, Russia, 1998.

[Roussel et al.98b] Roussel (David) et Pernel (Didier). – Int gration de pr dictions linguistiques dans un système de reconnaissance de la parole : une expérience utilisant une grammaire d'arbres lexicalisés. *Conférence sur le Traitement Automatique du Langage Naturel (TALN'98).* – 1998.

[Roussel et al.99a] Roussel (David), Kurdi (Zakaria M.) et Caelen (Jean). – Normalisation des extragrammaticalités, supertagging et analyse partielle pour le traitement de la parole. *Atelier TALN sur les méthodes hybrides pour le traitement robuste de la langue, Cargèse.* – 1999.

[Roussel et al.99b] Roussel (David) et Lopez (Patrice). – Contribution à l'analyse robuste non-déterministe pour les systèmes de dialogue parlé. *TALN'99, Cargèse.* – 1999.

[Roussel99] Roussel (David). – *Intégration de l'analyse du langage naturel parlé avec la reconnaissance de la parole.* – Thèse de PhD, Université Joseph Fourrier, 1999.

[Rubin87] Rubin (D.L.). – Divergence and convergence between oral and written communication. *Topics in Language Disorders*, vol. 7, 1987, pp. 1–18.

[Sabah et al.97] Sabah (G.), Vivier (J.), Pierrel (J.M.), Romary (L.), Vilnat (A.) et Nicolle (A.). – *Machine, langue et dialogue.* – L'Harmattan, Paris, 1997.

[Sabah89] Sabah (Gérard). – *L'intelligence artificielle et le langage, vol 2 : Processus de compréhension.* – Hermès, Paris, 1989.

[Sarkar et al.96] Sarkar (Anoop) et Joshi (Aravind). – Coordination in tree adjoining grammars : Formalization and implementation. *COLING'96, Copenhagen*, pp. 610–615. – 1996.

[Sarkar96] Sarkar (Anoop). – Incremental parser generation for tree adjoining grammars. *Student session, ACL'96, Santa Cruz.* – 1996.

[Satta et al.89] Satta (Giorgio) et Stock (Oliviero). – Formal properties and implementation of bidirectional charts. *Eleventh International Joint Conference on Artificial Intelligence, Detroit, MI*, pp. 1480–1485. – 1989.

[Satta et al.91] Satta (Giorgio) et Stock (Oliviero). – A tabular method for island-driven context free language parsing. *Ninth Conference of the American Association for Artificial Intelligence, Anaheim*, pp. 143–148. – 1991.

[Satta et al.98] Satta (Giorgio) et Schuler (William). – Restrictions on tree adjoining languages. *COLING/ACL'98, Montréal, Quebec.* – 1998.

[Satta94] Satta (Giorgio). – Tree adjoining grammar parsing and boolean matrix multiplication. *Computational Linguistics*, vol. 20, n° '27 2, 1994, pp. 173–192.

[Sauvage92] Sauvage (Caroline). – *Parallélisme et traitement automatique des langues, application à l'analyse des énoncés elliptiques.* – Thèse de PhD, Université Paris XI Orsay, 1992.

[Schabes et al.88] Schabes (Yves) et Joshi (Aravind K.). – An earley-type parsing algorithm for tree adjoining grammars. *ACL'88, Buffalo.* – 1988.

[Schabes et al.90a] Schabes (Yves) et Joshi (Aravind K.). – *Parsing Lexicalized Grammars.* – M. Tomita (ed), Current Issues in Parsing Technologies, Kluwer, 1990.

[Schabes et al.90b] Schabes (Yves) et Vijay-Shanker (K.). – Deterministic left to right parsing of tree adjoining languages. *ACL'90, Pittsburg.* – 1990.

[Schabes et al.94] Schabes (Yves) et Shieber (Stuart). – An alternative conception of tree-adjoining derivation. *Computational Linguistics*, vol. 20, n° '27 1, 1994, pp. 91–124.

[Schabes et al.95a] Schabes (Yves) et Waters (R. C.). – Tree insertion Grammar : A Cubic-Time, Parsable Formalism that Lexicalizes Context-Free Grammar without Changing the Trees Produced. *Computational Intelligence*, vol. 21, 1995, pp. 479–514.

[Schabes et al.95b] Schabes (Yves) et Waters (Richard). – Tree inserted grammar : A cubic-time parsable formalism that lexicalizes context free grammar without changing the trees produced. *Computational Linguistics, MIT Press*, 1995.

[Schabes92] Schabes (Yves). – Stochastic Lexicalized Tree Adjoining Grammars. *COLING.* – Nantes, France, 1992.

[Schabes94] Schabes (Yves). – Left to Right Parsing of Lexicalized Tree Adjoining Grammars. *Computational Intelligence*, vol. 10, 1994, pp. 506–524.

[Schank75] Schank (Roger C.). – *Conceptual Information Processing.* – North-Holland Publishing Company, 1975.

[Schuler98] Schuler (William). – Exploiting semantic dependencies in parsing. *International Workshop TAG+4, Philadelphia.* – 1998.

[Schwartz et al.96] Schwartz (R.), Miller (S.), Stallard (D.) et Maakhoul (J.). – Language understanding using hidden understanding models. *ICSLP'96, Philadelphia.* – 1996.

[SD96] Saint-Dizier (Patrick). – *Verb Semantic Classes in French.* – Rapport technique, IRIT-CNRS, draft, octobre 1996.

[Selkirk84] Selkirk (E.). – *Phonology and Syntax.* – MIT Press, Cambridge, 1984.

[Seneff86] Seneff (S.). – A computational model for the peripheral au ditory system : Application to speech recognition research. *ICASSP '86.* – 1986.

[Seneff92] Seneff (Stephanie). – A relaxation method for understanding spontaneous speech utterances. *Speech and Natural Language Workshop, San Mateo, CA.* – 1992.

[Sfez90] Sfez (Lucien). – *Critique de la communication.* – Seuil, Paris, 1990.

[Shieber et al.90] Shieber (Stuart) et Schabes (Yves). – Synchronous Tree Adjoining Grammars. *COLING*, pp. 253–260. – Helsinki, 1990.

[Shieber et al.92] Shieber (Stuart M.) et Schabes (Yves). – Generation and synchronous tree-adjoining grammars. *Computational Intelligence*, vol. 7, n° '27 4, 1992, pp. 220–228.

[Shieber et al.95] Shieber (Stuart), Schabes (Yves) et Pereira (Fernando). – Principles and Implementation of Deductive Parsing. *Journal of Logic Programming*, vol. 24, 1995, pp. 3–36.

[Shieber83] Shieber (Stuart). – Sentence disambiguation by a shift-reduce parsing technique. *IJCAI'83*, pp. 699–703. – 1983.

[Shieber85] Shieber (Stuart). – Evidence against the context-freeness of natural languages. *Linguistics and Philosophy*, vol. 8, n° '27 3, 1985, pp. 333–345.

[Shieber86] Shieber (Stuart). – An introduction to unification-based approaches to grammar. *Lectures Notes Number 4.* – CSLI, Stanford, CA, 1986.

[Shieber92] Shieber (Stuart). – *Constraint-Based Grammar Formalisms.* – MIT Press, Cambridge, Massachusetts, 1992.

[Shieber94] Shieber (Stuart). – Restricting the weak-generative capacity of synchronous tree-adjoining grammars. *Computational Intelligence*, vol. 10, 1994, pp. 371–385.

[Shriberg94] Shriberg (E. E.). – *Preliminaries to a Theory of Speech Disfluencies.* – Thèse de PhD, University of California, Berkeley, 1994.

[Sikkel et al.93] Sikkel (Klaas) et op den Akker (Rieks). – Predictive Head-Corner chart parsing. *Third International Workshop on Parsing Technologies.* – Tilburg, Netherlands, 1993.

[Sikkel97] Sikkel (Klaas). – Parsing schemata - a framework for specification and analysis of parsing algorithms. *Texts in Theoritical Computer Science.* – EATCS Series - Springer-Verlag, Berlin/Heidelberg/New York, 1997.

[Simon52] Simon (H. A.). – A formal theory of interaction in social groups. *American Sociological Review*, vol. 17, 1952, pp. 202–211.

[Smaili91] Smaïli (Kamel). – *Conception et réalisation d'une machine à dicter à entrée vocale destinée aux grands vocabulaires : le système MAUD.* – Thèse de PhD, Université de Nancy 1, 1991.

[Smedt et al.90] Smedt (K. De) et Kempen (G.). – Segment grammar : a formalism for incremental generation. *Natural language generation and computational linguistics.* – Kluwer, 1990.

[Smets98] Smets (Martine). – Comparison of xtag and lexsys grammars. *Fourth International Workshop on Tree Adjoining Grammars (TAG+4), Philadelphia.* – 1998.

[Souvay92] Souvay (Gilles). – *DIAPASON, un environnement de développement pour l'intégration d'une entrée vocale dans des applications de type commande de machine.* – Thèse de PhD, Université de Nancy 1, 1992.

[Sperber et al.86] Sperber (Dan) et Wilson (Deidre). – *Relevance, Communication and Cognition.* – Blackwell, 1986.

[Srinivas et al.95] Srinivas (Bangalore), Doran (Christine) et Kulick (Seth). – Heuristics and parse ranking. *IWPT'95, Prague.* – 1995.

[Srinivas97a] Srinivas (Bangalore). – *Complexity of lexical descriptions and its relevance to partial parsing.* – Thèse de PhD, University of Pennsylvania, Philadelphia, 1997.

[Srinivas97b] Srinivas (Bengalore). – Performance evaluation of supertagging for partial parsing. *International Workshop on Parsing Technology (IWPT'97).* – 1997.

[Strom97] Ström (Nikko). – *Automatic continuous speech recognition with rapid speaker adaptation for human/machine interaction.* – Thèse de PhD, KTH, Stockholm, 1997.

[Tannen85] Tannen (Deborah). – Relative focus on involvement in oral and written discourse. *Literacy, language and learning : The nature and consequence of reading and writing*, éd. par Olson (D.R.), Torrance (N.) et Hildyard (A.), pp. 124–147. – Cambridge University Press, 1985.

[Tannenbaum et al.65] Tannenbaum (P.), Williams (F.) et Hillier (C.). – Word predictability in the environments of hesitation. *Journal of Verbal Learning and Verbal Behavior*, vol. 4, 1965.

[Tesniere59] Tesnière (Lucien). – *Éléments de syntaxe structurale.* – Klincksieck, Paris, 1959.

[Tomabechi91] Tomabechi (Hideto). – Quasi-destructive graph unification. *29th Meeting of the Association for Computational Linguistics, Berkeley, CA*, pp. 315–322. – 1991.

[Tomita87] Tomita (Masaru). – An efficient augmented context-free parsing algorithm. *Computational Linguistics*, vol. 13, n° 1, 1987, pp. 31–46.

[vN et al.97] van Noord (Gertjan) et Bouma (Gosse). – Hdrug. A flexible and Extendible Development Environment for Natural Language Processing. *ENVGRAM Workshop.* – Madrid, Spain, 1997.

[vN et al.98] van Noord (Gertjan), Bouma (Gosse), Koeling (Rob) et Nederhof (Mark-Jan). – Robust Grammatical Analysis for Spoken Dialogue Systems. *Natural Language Engineering*, vol. 1, 1998, pp. 1–48.

[vN94] van Noord (Gertjan). – Head Corner Parsing for TAG. *Computational Intelligence*, vol. 10, 1994, pp. 525–534.

[vN97] van Noord (Gertjan). – An Efficient Implementation of the Head-Corner Parser. *Computational Linguistics*, 1997.

[VS et al.88] Vijay-Shanker (K.) et Joshi (A.). – A feature-based tree adjoining grammar. *12th COLING, Budapest.* – 1988.

[VS et al.92] Vijay-Shanker (K.) et Schabes (Yves). – Structure sharing in lexicalized tree adjoining grammars. *COLING'92, Nantes.* – 1992.

[VS et al.93] Vijay-Shanker (K.) et Weir (David J.). – Parsing some constrained grammar formalisms. *Computational Linguistics*, vol. 19, 1993, pp. 591–636.

[VS87] Vijay-Shanker (K.). – *A Study of Tree Adjoining Grammar.* – Thèse de PhD, University of Pennsylvania, Philadelphia, 1987.

[VS93] Vijay-Shanker (K.). – Using desciptions of trees in tree adjoining grammars. *Computational Linguistics*, vol. 18, n° '27 4, 1993.

[Vygotsky87] Vygotsky (Lev). – *Thought and Language.* – The MIT Press, 1987.

[vZ et al.99] van Zanten (Gert Veldhuijzen), Bouma (Gosse), Sima'an (Khalil), van Noord (Gertjan) et Bonnema (Remko). – Evaluation of the nlp components of the ovis2 spoken dialogue system. *CLIN'99.* – 1999.

[Webber et al.98] Webber (Bonnie Lynn) et Joshi (Aravind K.). – Anchoring a Lexicalized Tree-Adjoining Grammar for Discourse. *COLING.* – COLING-ACL'98 Workshop on Discourse Relations and Discourse Markers, 1998.

[Wehrli97] Wehrli (E.). – *L'analyse syntaxique des langues naturelles : problèmes et méthodes.* – Masson, Paris, 1997.

[Weir88] Weir (David). – *Characterizing Midly Context-Sensitive Grammar Formalisms.* – Thèse de PhD, University of Pennsylvania, 1988.

[Westby85] Westby (C.E.). – Learning to talk – talking to learn : Oral-literate language differences. *Comminication skills and classroom success : Therapy methodologies for language-learning disabled students*, éd. par Simon (C.S.), pp. 181–218. – College-Hill Press, 1985.

[Wolf et al.80] Wolf (J. J.) et Wood (W. A.). – *The HWIM Speech Understanding System.* – Englewood Cliffs, N.J. : Prentice-Hall Inc, 1980.

[Wolff99] Wolff (Frédéric). – *Analyse contextuelle des gestes de désignation en dialogue homme-machine.* – Thèse de PhD, Université Henri Poincaré Nancy 1, 1999.

[Woods70] Woods (W.A.). – Transition network grammars for natural language analysis. *Communication of the ACM*, vol. 13, n° '27 10, 1970, pp. 591–596.

[Younger67] Younger (D. H.). – Recognition and parsing of context-free language in cubic time. *Information and Control*, vol. 10, n° '27 2, 1967, pp. 189–208.

[Zue et al.89] Zue (V. W.), Glass (J.) et Seneff (S.). – Acoustic segmentation and phonetic classification in the summit system. *Proceedings Int. Conf. on Acoustics, Speech and Signal Processing.* – 1989.

université HENRI POINCARÉ
NANCY 1

FACULTÉ DES SCIENCES

Monsieur **LOPEZ Patrice**

DOCTORAT de l'UNIVERSITE HENRI POINCARE, NANCY-I

en INFORMATIQUE

VU, APPROUVÉ ET PERMIS D'IMPRIMER

Nancy, le 25 octobre 1999 n° 265

Le Président de l'Université

UNIVERSITÉ HENRI POINCARÉ, NANCY
LE PRÉSIDENT
CL. BURLET

3C-F9

Résumé

Cette thèse est une contribution à l'analyse grammaticale d'énoncés oraux dans le cadre d'un système de dialogue homme-machine. Le gain potentiel pour une application se situe à la fois en termes de performance, de robustesse et de couverture de la langue, mais également en termes d'interaction avec un module de reconnaissance de la parole. Nous nous sommes fondé sur le formalisme des grammaires lexicalisées d'arbres adjoints (LTAG), choix que nous justifions, à la vue des différents formalismes existants, par des propriétés intéressantes autant linguistiques qu'informatiques. Afin d'adapter ce formalisme à l'analyse locale et robuste d'énoncés oraux, nous présentons tout d'abord un algorithme original délivrant des analyses pour les différents segments grammaticaux présents dans l'énoncé. Cet algorithme repose sur des techniques tabulaires et de compaction de la grammaire ce qui associe efficacité et robustesse. De manière complémentaire, nous présentons des principes permettant de capturer les contraintes de la langue parlée avec une LTAG, ainsi que des mécanismes additionnels de détection et de réparation de certaines distorsions syntaxiques de l'oral. L'ensemble de ces propositions est intégré dans un système implanté nommé EGAL (Extraction de Grammaire d'Arbres Lexicalisée), permettant de spécialiser de façon semi-automatique une grammaire générale de la langue à un sous-langage d'application spécifique à l'aide de corpus de type Magicien d'Oz. Le système fournit également un atelier de tests d'analyse pour évaluer la grammaire obtenue et les solutions robustes proposées.

Mots-clés: LTAG, analyse robuste de l'oral, Dialogue Homme Machine.

Abstract

This thesis investigates a linguistically motivated grammatical formalism for the robust parsing of spoken utterances in the context of Man-Machine Dialogue. We propose a framework for the spoken language processing based on Lexicalized Tree Adjoining Grammars (LTAG). LTAG is an interesting formalism for both parsing and linguistic descriptions. In order to adapt the formalism to spoken dialogue, we present first a parsing algorithm that delivers derivations for all the most extended grammatical chunks of an utterance - the connection driven parsing algorithm. The efficiency of this algorithm is based on tabular techniques and structure sharing precompilations. Complementary, we propose principles to capture spoken language constraints in a classical LTAG and additional parsing repairs rules dedicated to phenomena as spoken disfluencies. These propositions are implemented in a system called EGAL (Lexicalized Tree Grammars Extraction) which allows an assisted specialization of a general grammar to an application sublanguage according to a Wizard of Oz corpora. The system allows also to test various parsing algorithms in order to evaluate the resulting grammar and our robust algorithms.

Keywords: LTAG, robust parsing, Spoken Dialogue System.