Université Paris 7
UFR d'informatique
Laboratoire de Traitement Automatique du Langage Naturel (TALANA)

Un modèle de reformulation automatique fondé sur la Théorie Sens-Texte

Application aux langues contrôlées

Alexis Nasr

Thèse de Doctorat d'informatique

présentée le 19 novembre 1996

Jury :

Anne Abeillé	Examinateur
Laurence Danlos	Directeur
Corinne Fournier	Examinateur
Catherine Fuchs	Rapporteur
Richard Kittredge	Rapporteur

Thèse préparée au sein de l'équipe d'ingénierie linguistique et documentaire
Etudes Avancées, Dassault Aviation - 78, quai Marcel Dassault - Cedex 300 - 92552 Saint Cloud

Table des matières

Resumé

La thèse propose un modèle de reformulation automatique de phrases basé sur la théorie linguistique Sens-Texte. Le modèle est appliqué au domaine des langues contrôlées, nées dans le milieu documentaire industriel et qui imposent, à certains types de textes techniques, des contraintes diverses parmi lesquelles des contraintes lexicales et syntaxiques. Le modèle proposé repose sur la théorie Sens-Texte, qui accorde une place importante à la paraphrase. L'idée principale de la thèse est d'utiliser les capacités paraphrastiques de la théorie pour réaliser la reformulation. Celle-ci repose sur l'identification, dans la grammaire et dans le lexique, des structures autorisées et des structures interdites. La reformulation, elle-même, consiste à remplacer, dans la représentation syntaxique d'une phrase, les structures interdites par des structures autorisées paraphrastiques grâce au système de paraphrase. Le travail proposé porte principalement sur l'intégration de la théorie Sens-Texte dans un cadre de reformulation automatique. Une telle intégration nécessite une réorganisation des connaissances au sein de la théorie dans le double but de proposer un moyen de représentation des structures lexico-syntaxiques interdites et autorisées, tout en optimisant le traitement automatique. Le modèle proposé comporte trois phases principales, une phase d'analyse à l'issue de laquelle une représentation syntaxique profonde de la phrase à reformuler est construite. Cette dernière sera modifiée lors de la phase de transformation par application de règles de paraphrases afin d'éliminer certaines constructions interdites. La dernière phase, dite de regénération, consiste à synthétiser une nouvelle phrase en s'assurant de ne pas réintroduire de structures interdites.

Abstract

We define a model of sentence automatic reformulation based on the Meaning-Text Theory (MTT). The model is applied to controlled languages, which impose several kinds of constraints on technical texts, among which lexical and syntactic ones. The system detects, in a sentence, structures which are not controlled language compliant then proposes an authorised paraphrase of the original sentence. The main idea of the thesis is to take advantage of the MTT paraphrastic system in order to perform reformulation. The lexicon and the grammar of the system distinguish between compliant and non compliant lexical and syntactic structures. The reformulation process replaces, in the syntactic representation of a sentence, non compliant structures by compliant ones by applying MTT paraphrastic rules. The work described focusses on the integration of MTT in an automatic reformulation framework. In order to perform this integration, some changes have been made on the MTT overall organisation. These changes aim at defining a way to describe authorised and non authorised lexical and syntactic structures while optimizing computer processing. The system is decomposed into three main components, a parser, which builds a deep syntactic structure of the sentence processed, a transformation component, which transforms the deep syntactic structure in order to eliminate some unauthorised structures and, eventually, a regeneration component, which builds a new sentence, free of non compliant structures.

Remerciements

J'adresse mes sincères remerciements à :

Laurence Danlos, mon directeur de thèse, pour la confiance sans faille qu'elle m'a temoigné depuis le début. Son soutien constant et ses conseils éclairés, jamais dogmatiques, m'ont beaucoup aidé tout au long de ces trois années.

Corinne Fournier qui m'a accueilli au sein de l'équipe de traitement du langage naturel de Dassault Aviation, m'a initié à la Théorie Sens-Texte et à élaboré avec moi le sujet de cette thèse.

Jean Saulais, chef du département d'intelligence artificielle et d'informatique avancée de Dassault Aviation, qui m'a permi d'effectuer cette thèse dans d'excellentes conditions.

Catherine Fuchs et Richard Kittredge qui ont accepté d'être rapporteurs de cette thèse et m'ont apporté un regard neuf, me permettant de prendre de la distance par rapport à mon travail.

Anne Abeillé pour l'intêret qu'elle a toujours porté à mon travail et pour avoir bien voulu jouer le rôle d'examinateur.

Farid Cerbah qui m'a accompagné depuis le début de ma thèse tant d'un point de vue scientifique que sur le plan humain. Les longues et nombreuses discussions que nous avons eues tout au long de ce parcours m'ont aidées à en venir à bout.

Owen Rambow qui a assisté, durant son année parisienne, à la genèse de la thèse et m'a poussé à développer des idées qui me paraissaient, à l'époque, bien perilleuses.

Olivier Sigaud pour son aide précieuse lors de la mise au point de la partie informatique de mon projet. Ses remarques stimulantes m'ont conduit à de nombreuses améliorations tant pratiques que théoriques.

Sylvain Kahane qui s'est montré un interlocuteur précieux et m'a incité à formaliser plusieurs notions, encore à l'état d'esquisses avant son intervention.

Je voudrais enfin rappeler mon amitié aux membres de l'équipe d'ingénierie linguistique et documentaire de Dassault Aviation ainsi qu'à ceux du laboratoire TALANA grâce auxquels ces années resteront chères à ma memoire.

Abréviations

TAL	Traitement Automatique de la Langue
TST	Théorie Sens-Texte
MST	Modèle Sens-Texte
DEC	Dictionnaire Explicatif Combinatoire
FL_{syn}	Fonction Lexicale Syntagmatique
FL_{par}	Fonction Lexicale Paradigmatique
RMorphP	Représentation Morphologique Profonde
RSyntP	Représentation Syntaxique Profonde
RSyntS	Représentation Syntaxique de Surface
SSyntP	Structure Syntaxique Profonde
SSyntS	Structure Syntaxique de Surface
AEP	Arbre Elémentaire Profond
AES	Arbre Elémentaire de Surface
AES+	Arbre Elémentaire de Surface Etendu
AEP+	Arbre Elémentaire Profond Etendu
$\mathcal{AES}$	Type d'Arbre Elémentaire de Surface
$\mathcal{AEP}$	Type d'Arbre Elémentaire Profond
$\mathcal{AES}+$	Type d'Arbre Elémentaire de Surface Etendu
$\mathcal{AEP}+$	Type d'Arbre Elémentaire Profond Etendu
TAG	Tree Adjoining Grammar
LTAG	Lexicalised Tree Adjoining Grammar

Présentation

Les travaux décrits dans cette thèse procèdent d'un but tout à fait pragmatique : il s'agit de déceler, dans les lignes des manuels d'entretien et de fonctionnement de systèmes aéronautiques, des formulations proscrites par une langue contrôlée, pour en proposer d'autres, conformes à la norme. Un tel programme, par la diversité des problèmes auxquels il s'attaque, dépasse le cadre d'une thèse, c'est pourquoi les quelques lignes qui suivent nous semblent nécessaires pour préciser notre position vis-à-vis de ce programme. Celle-ci est fondée principalement sur les deux choix fondamentaux suivants.

Le premier consiste en une volonté de conserver une vision globale du processus de reformulation. Les difficultés auxquelles est confronté un système de reformulation sont en effet pléthoriques. On pourra citer, entre autres : la définition généralement incomplète des règles des langues contrôlées, les erreurs orthographiques, les ambiguïtés ou les constructions atypiques des phrases à reformuler, les glissements sémantiques ou les malformations syntaxiques apparaissant lors de la reformulation ... Un système fiable de reformulation doit pouvoir faire face à tous ces problèmes qui relèvent encore, pour la plupart d'entre eux, de la recherche. Deux attitudes sont alors possibles pour qui s'intéresse à la reformulation : une première consiste à se focaliser sur un de ces problèmes au détriment de la vision d'ensemble ; la seconde, que nous avons adoptée dans cette thèse, cherche plutôt à proposer un modèle complet de reformulation quitte à laisser un certain nombre de problèmes non traités. Cette vision globale permet de prendre en compte, lors de l'élaboration du modèle, des contraintes propres au processus complet, qui n'apparaîssent pas dans le cadre d'études plus locales portant sur un seul aspect du processus. Les différents éléments du modèle que nous allons proposer sont en effet définis en fonction des différentes étapes du processus de reformulation. Nous proposerons en particulier un formalisme de représentation des connaissances lexico-syntaxiques qui, tout en étant conçu pour les différentes phases de la reformulation, notamment l'analyse et la génération, permet de distinguer dans la grammaire et le lexique les constructions interdites et autorisées par une langue contrôlée.

Le second choix consiste à nous placer d'emblée dans le cadre d'une théorie linguistique. La reformulation est en effet avant tout une tâche linguistique que l'on ne peut prétendre traiter, dans une thèse d'informatique, hors d'un cadre linguistique théorique. La théorie que nous avons choisi d'adopter est la Théorie Sens-Texte (notée dorénavant *TST*) : celle-ci présente, en partie grâce à son système de paraphrase, des atouts importants pour la reformulation. Le modèle de reformulation que nous proposons épouse la majorité des hypothèses linguistiques de la *TST*, notamment l'adoption des structures de dépendances pour les représentations syntaxiques et le recours aux différents niveaux de représentation des énoncés qu'elle définit. La *TST* possède néanmoins, à l'instar de toute théorie linguistique, des défauts, des imprécisions et des manques. Ceux-ci peuvent par moment nuire à la mise en œuvre de la tâche de reformulation et nous serons amenés, dans deux types de cas à nous éloigner de la théorie. Le premier cas se présentera lorsque l'organisation interne de la *TST* se révèlera inadaptée à la mise en œuvre de la reformulation automatique, comme nous le verrons notamment lors de l'analyse syntaxique. Nous proposerons alors une organisation alternative des connaissances, visant à une mise en œuvre informatique plus efficace. Le second cas se présentera lorsque les opérations définies par la *TST* conduiront à la génération de reformulations manifestement fausses, conséquence des imperfections du système de paraphrase. Nous proposerons dans ce cas précis des modifications à apporter au système de paraphrase tel qu'il est conçu dans la *TST* ainsi qu'au niveau de représentation syntaxique profond, auquel s'applique le système de paraphrase. Seules certaines de ces modifications seront intégrées dans notre modèle. D'autres, remettant en cause des principes fondamentaux de la *TST*, ne seront qu'évoquées.

Après avoir clarifié notre position, il reste à souligner que le travail décrit dans cette thèse est avant tout un travail de modélisation qui propose un modèle implémentable de reformulation basé sur la *TST*. L'effort fourni porte plus sur la conception du modèle que sur le nombre et la variété de phénomènes linguistiques traités. Nous proposerons en annexe B la grammaire et les règles de paraphrase que nous avons implémentés pour illustrer et tester le modèle. L'implémentation du modèle a été effectuée en LISP. Nous ne décrirons pas ici le plan de la thèse que l'on pourra retrouver à la fin du premier chapitre, après une introduction détaillée de notre problématique.

Chapitre 1

Introduction : reformulation dans une langue contrôlée

La conception d'un système de reformulation nécessite de connaître quels types de textes reformuler et quels types de reformulations effectuer. Nous allons nous pencher successivement sur ces deux points dans les deux premières parties de ce chapitre. Nous commencerons par aborder la problématique des *langues de spécialité* dont les particularités par rapport à la langue standard en font des candidats intéressants pour le traitement automatique. Nous passerons en revue différentes études que les langues de spécialité ont suscitées et les conclusions auxquelles elles ont abouti. Nous nous demanderons alors si la langue à laquelle nous nous intéressons dans le cadre de cette thèse peut être considérée comme une langue de spécialité. Le type de reformulation que nous cherchons à effectuer est, lui, guidé par la problématique des *langues contrôlées* qui sera présentée dans la seconde partie de ce chapitre. Les langues contrôlées, nées dans le cadre de la documentation industrielle, imposent un certain nombre de contraintes stylistiques à des documents techniques. Le but de la reformulation est de remplacer dans une phrase les structures non conformes à une langue contrôlée par des constructions qui le sont. Nous décrirons la problématique des langues contrôlées et analyserons les problèmes que leur prise en compte dans des applications informatiques rencontrent. La problématique de la thèse sera alors introduite, nous définirons en particulier le rôle joué par une théorie linguistique, ici la Théorie Sens-Texte, dans le processus de reformulation. Le chapitre s'achèvera sur une présentation du plan de la thèse.

1.1 Langues de spécialité

Nous nous intéresserons dans le cadre de cette thèse à des textes de maintenance aéronautique. Ces derniers décrivent d'une part les systèmes mécaniques, hydrauliques et électriques des avions et indiquent d'autre part la marche à suivre pour effectuer les opérations d'entretien nécessaires à la bonne marche des systèmes. Ces textes présentent plusieurs particularités lexico-syntaxiques qui sont le reflet de domaines de connaissance limités (l'aéronautique, la mécanique, l'hydraulique et l'électrique) et de buts communicatifs précis (décrire le fonctionnement des systèmes physiques et leur entretien). Ce type de textes, fortement lié à un domaine technique, a attiré l'attention de linguistes d'horizons différents qui leur ont donné pour nom, selon les écoles, langue de spécialité, sous-langues (sublanguages), langues techniques ou « *language for special purposes* ». Nous avons opté ici pour le terme *langues de spécialité* qui, contrairement au terme *sous-langue*, met en avant les deux aspects de ce phénomène que sont d'une part la langue et d'autre part un domaine de connaissances. Les nombreuses études portant sur les langues de spécialité se sont intéressées à différents aspects de ce phénomène. Selon les buts poursuivis par leurs auteurs, ces derniers ont apporté des éclairages différents sur la notion de langue de spécialité. Un premier classement possible consiste à regrouper d'une part des travaux *didactiques*, dont le but est l'apprentissage d'une langue de spécialité par un rédacteur, et d'autre part des travaux *descriptifs*, dont le but est de cerner et de décrire les langues de spécialité sans chercher à les façonner ou à enseigner leur usage. Les résultats des études didactiques se présentent sous la forme de guides à l'usage des rédacteurs. Ces guides se composent de règles de rédaction codifiant un certain style, adapté aux besoins d'une communication particulière. Ils prônent généralement la clarté et la précision au dépend de l'imagination ou de l'inventivité et sont de ce point de vue fortement normatifs. On trouvera dans cette catégorie des ouvrages tels que [Wieringa et al., 1992] ou [Cunningham & Cohen, 1984]. Nous reviendrons plus en détail sur ce type d'approche en 1.2.

Les études composant la seconde famille partagent un même souci de description du phénomène linguistique des langues de spécialité. Néanmoins, elles ne constituent pas une famille homogène et une distinction peut être opérée entre deux groupes. Le premier est constitué d'études de chercheurs, souvent européens, visant à élaborer une « *linguistique de la langue savante* » ([Lerat, 1995]) puisant ses fondements dans la linguistique générale. Ces travaux cherchent à définir la place des langues de spécialité dans divers systèmes linguistiques et sémiotiques. On trouvera dans cette catégorie [Lerat, 1995] ou [Kocourek, 1991]. Le second groupe est constitué de chercheurs, principalement nord Américains, s'inscrivant dans la suite des travaux de [Harris, 1968] et visant à l'élaboration de modèles formels de description des langues de spécialité. Les langues de spécialité sont vues ici comme des sous-ensembles de la langue standard. Le rôle du linguiste est alors d'étudier leurs régularités et les restrictions qu'elles présentent vis-à-vis de la langue standard dans le but de les décrire sous la forme de grammaires. On trouvera les résultats de nombreuses études dans [Kittredge, 1982] et [Grishman & Kittredge, 1986]. Les descriptions lexico-syntaxiques très précises résultant de ces travaux sont généralement utilisées dans le cadre d'applications informatiques que nous décrirons brièvement en 1.1.2.

1.1.1 Délimitation d'une langue de spécialité

Le point de vue théorique sur les langues de spécialité proposé par Z. Harris où une langue de spécialité est vue comme un « *sous-ensemble des phrases d'une langue clos pour des opérations définies par la langue* » ne fait pas référence à la réalité sociologique de ce phénomène. Une langue de spécialité, ou sous-langue selon la terminologie de Z. Harris, est un phénomène naturel issu de l'emploi de la langue dans des conditions particulières de communication. Indépendamment de la volonté de modéliser les langues de spécialité, se pose le problème de leur délimitation. Quels critères prendre en compte pour décider de regrouper plusieurs textes sous l'étiquette d'une langue de spécialité particulière ? Le choix n'est pas toujours aisé à effectuer, nombreux sont les cas où des textes partagent un certain nombre de caractéristiques tout en se distinguant pour d'autres. Le but est d'arriver à cerner un ensemble qui soit linguistiquement homogène et qui corresponde à des corpus existants. Le risque est d'une part d'adopter une attitude trop permissive et de considérer un ensemble hétérogène duquel n'émergent pas de caractéristiques communes importantes et d'autre part, d'adopter une attitude trop restrictive aboutissant à ne garder qu'un ensemble de textes ou de portions de texte trop restreint pour être représentatif. Nous avons essayé ici de regrouper plusieurs critères apparaissant dans [Kittredge, 1982, Lehrberger, 1982] et [Kocourek, 1991] pouvant être pris en compte pour délimiter une langue de spécialité.

- La langue de référence

  Le critère le plus immédiat de classement d'une langue de spécialité est évidemment la langue dans laquelle elle s'inscrit. Bien qu'une langue de spécialité puisse par moment s'éloigner de la langue standard, elle continue toujours à puiser dans les ressources linguistiques de cette dernière, de sorte qu'un usager du français, par exemple, pourra, face à un texte d'une langue de spécialité et sans rien connaître au domaine technique auquelle elle se réfère, décider s'il s'agit d'un texte en français.

- Le domaine de référence

  Une langue de spécialité se définit souvent par rapport à un domaine sémantique : c'est ainsi que l'on parle du français de la physique quantique ou de l'anglais de la biologie moléculaire, mettant l'accent sur le domaine de référence de la langue de spécialité. Ce dernier influence les caractéristiques linguistiques de la langue de spécialité et en particulier son lexique. Les études menées dans le cadre du projet TAUM-AVIATION ([Lehrberger, 1982]) ont estimé la taille du lexique des textes de maintenance d'avions rédigés en anglais à 40.000 unités, chiffre qu'il convient de mettre en rapport avec les 450.000 entrées d'un dictionnaire général de l'anglais. Les restrictions lexicales concernent principalement les catégories verbales, nominales, adverbiales et adjectivales. Les mots grammaticaux sont par contre peu affectés par ces restrictions. En ce qui concerne la polysémie, l'influence du domaine de référence est là aussi importante, permettant la désambiguïsation de certains mots. On pourra citer l'exemple du verbe *voler*, dont un seul sens peut être retenu dans le cadre des manuels de maintenance aéronautique.

- La visée communicative des textes

Le choix d'un domaine sémantique et d'une langue de référence particuliers ne permettent pas à eux seuls de délimiter une langue de spécialité particulière. Les textes des manuels de maintenance aéronautique en offrent un bon exemple. Bien que partageant un domaine sémantique identique, différentes parties de tels manuels peuvent présenter des différences importantes selon qu'elles décrivent le fonctionnement d'un système ou la marche à suivre pour effectuer une procédure d'entretien. La définition d'une langue de spécialité se doit par conséquent de prendre en compte en sus d'un domaine sémantique particulier, la fonction des textes, qui va influencer la syntaxe de la langue de spécialité. [Lehrberger, 1982] note, entre autres, l'absence de formes interrogatives directes et de verbes au passé dans les textes de maintenance aéronautique. Il ne s'agit pas pour autant d'une syntaxe simple : les constructions syntaxiques sont variées et les phrases longues et complexes sont fréquentes.

- La communauté des usagers

  Une langue de spécialité est un outil de communication partagé par les membres d'une certaine communauté. Dans le cas des manuels d'entretien aéronautiques, la communauté des rédacteurs et des lecteurs est assez bien délimitée. Ce n'est pas toujours le cas : [Kittredge, 1982] note l'exemple de la langue des rapports boursiers pour lesquels la communauté des usagers est difficile à cerner. Selon le niveau de connaissance des lecteurs et des rédacteurs, ces textes présentent des différences tant dans leur vocabulaire que dans la complexité de leur syntaxe. D'une manière générale, plus la communauté faisant usage d'une langue de spécialité donnée est homogène plus la langue de spécialité sera facile à cerner.

On peut remarquer qu'au vu de ces critères, les textes auxquels nous nous intéressons peuvent être considérés sans hésitations comme relevant d'une langue de spécialité. Leur domaine sémantique est clairement identifié, comme le sont leurs but communicatifs et la communauté de leurs usagers.

1.1.2 Langues de spécialités et applications informatiques

Les études menées sur les langues de spécialité par les équipes nord américaines évoquées en 1.1 sont généralement effectuées en vue d'applications informatiques. Les régularités et restrictions que présentent les langues de spécialité sont mises à profit pour concevoir des applications informatiques difficilement envisageables pour la langue générale. D'importants projets d'analyse de langues de spécialités recourant aux techniques d'analyse distributionnelles et transformationnelle de Z. Harris ont vu le jour. Elles sont généralement associées à deux types d'applications informatiques, la traduction automatique et l'extraction automatique d'informations.

A l'Université de New-York, le projet *Linguistic String Project* ([Sager, 1982, Sager, 1986])qui a pris naissance dans les années soixante s'est attaché à décrire des textes traitant de pharmacologie dans le but d'automatiser l'extraction d'informations de tels textes. L'analyse des textes se situe à la frontière d'une analyse distributionnelle et d'une analyse plus conceptuelle. L'idée est de mettre à profit deux structurations sous-jacentes aux textes, une structuration linguistique et une structuration conceptuelle, pour dégager des grammaires mêlant contraintes grammaticales et contraintes sémantiques. L'analyse des textes consiste dans un premier temps à établir des classes de mots en fonction de leur comportement syntaxique et de leur homogénéité sémantique. C'est ainsi que sont distinguées les classes nominales LI (pour lipides) regroupant les substantifs *cholestérol, triglycérides...*, la classe OR (pour organe ou cellule) regroupant les noms ou goupes nominaux *foie, tissus périphériques...* et des classes verbales élaborées en fonction de la sémantique des verbes et de leurs contraintes de sous-catégorisation. Parallèlement, les phrases sont décomposées en phrases élémentaires qui sont à leur tour analysées selon les catégories syntaxico-sémantiques définies ci-dessus. Les schémas de phrases élémentaires ainsi obtenus sont alors mis en relation avec des« *structures informationnelles*» représentées sous forme de tableaux ou *formats*. Ces structures représentent le contenu informationnel des phrases libéré des aspects purement linguistiques. La définition d'une grammaire et la mise en correspondance des schémas de phrase permettent alors d'analyser automatiquement des textes du domaine de spécialité puis d'en *formatter* le contenu. Cette représentation peut alors servir de base pour différents traitements de l'information.

A l'Université de Montréal, dans le cadre du groupe TAUM (Traduction Automatique à l'Université de Montréal), de nombreuses langues de spécialité ont été étudiées en vue d'une application de traduction automatique. Les textes d'une langue de spécialité présentent un double avantage vis-à-vis de la traduction automatique. D'une part, leur lexique et leur syntaxe sont réduits par rapport à la langue générale, facilitant la phase d'analyse et d'autre part, comme l'observe [Kittredge, 1982], le français et l'anglais d'un même domaine de spécialité présentent des ressemblances importantes, simplifiant d'autant la phase de transfert. Plusieurs langues de spécialité dans leurs versions anglaises et françaises ont donc été étudiées, en commençant par la langue des rapports météorologiques en 1974, qui a donné lieu à un système de traduction automatique ([Chevalier, 1978]), la langue de maintenance aéronautique à partir de 1976 ([Isabelle, 1978, Lehrberger, 1982]), qui n'a toutefois pas abouti à un système de traduction automatique. D'autres langues de spécialité concernant, entre autres, la micro-économie et les recettes de cuisine ont ensuite été étudiées dans le cadre d'un projet de syntaxe contrastive qui a débuté en 1978 ([Kittredge, 1982]).

1.2 Langues contrôlées

Parallèlement à la problématique des langues de spécialité, s'est développée dans le milieu de la rédaction technique industrielle la notion de langues contrôlées. Ces dernières sont constituées de règles d'écriture imposant un certain nombre de contraintes à la rédaction de textes techniques. Les buts visés sont, d'une part d'accroître la lisibilité des textes afin de limiter les possibles erreurs d'interprétation et d'autre part, d'en faciliter la traduction. Les contraintes concernent aussi bien des aspects linguistiques (terminologie, syntaxe, pragmatique) que documentaires (formats des feuilles, pagination, polices de caractères ...). Le phénomène des langues contrôlées n'est pas sans rapport avec les approches didactiques plus conventionnelles que nous avions identifiées en 1.1 qui cherchaient elles aussi à codifier un certain style de rédaction, si bien que l'on peut se demander s'il s'agit de phénomènes différents. Bien que les deux phénomènes relèvent à notre avis d'une même problématique, les types de textes auxquels s'intéressent les langues contrôlées sont généralement mieux délimités que dans les approches conventionnelles : ils relèvent de domaines de spécialité très précis. Cette meilleure connaissance *a priori* des textes permet d'aller plus loin dans leur contrôle en imposant, en particulier, un vocabulaire, ce que ne peut pas faire une approche plus générale, dont le domaine sémantique est moins bien délimité. D'autre part, les langues contrôlées sont souvent la propriété industrielle de certaines firmes et constituent, dans certains cas, des exigences contractuelles, ce qui contribue à leur donner un statut particulier.

On pourra trouver dans [Adriaens & Schreurs, 1992] une lignée de langues contrôlées utilisée dans l'industrie en remontant à l'établissement, dans les années soixante, de l'*anglais fondamental* par la firme Caterpillar (Caterpillar Fundamental English). Par la suite, plusieurs entreprises industrielles ou corps de métier ont adopté la formule et de nombreuses langues contrôlées ont vu le jour. Dans les métiers de l'aéronautique, confrontés à des volumes documentaires prodigieux, un dérivé de l'anglais fondamental de Caterpillar, connu sous le nom d'*anglais simplifié* (Simplified English) a été élaboré en 1982 par l'association européenne des industries aéronautiques (AECMA) [AECMA, 1989]. Ce dernier a ensuite servi de modèle au *français rationalisé* développé par le groupement d'industries françaises aéronautiques et spatiales (GIFAS) [GIFAS, 1990]. C'est à notre connaissance le seul représentant d'une langue contrôlée pour le français.

Au niveau linguistique, qui est celui qui nous intéresse ici, le français rationalisé se présente sous la forme d'un vocabulaire contrôlé (glossaire) composé à l'heure actuelle d'un millier de mots et d'un ensemble de règles permettant « *d'utiliser ce vocabulaire* ». La formule est vague, nous verrons qu'il s'agit en fait d'un ensemble hétérogène de règles de niveaux linguistiques différents.

Du point de vue du lexique, qui constitue la partie linguistique la plus homogène du français rationalisé, trois familles de mots sont distinguées :

- Les mots du glossaire. Ils forment en quelque sorte la partie non-technique du français rationalisé et constituent le *vocabulaire général* des textes. On retrouve dans cette catégorie différents mots outils, tels que les conjonctions de subordination, conjonctions de coordination, prépositions ainsi que des verbes, substantifs, adjectifs et adverbes communs, tels que *enlever, serrer, roue...*

- Les termes techniques. Ce sont des substantifs et des adjectifs, trop nombreux pour qu'on puisse en fournir une liste complète. Ils sont néanmoins regroupés en vingt familles sémantiques : termes de navigation et de vol, infrastructures aéronautiques...

- Les termes relatifs à des procédés industriels. Ce sont des noms et des verbes, regroupés en six familles sémantiques : verbes indiquant un enlèvement de matière, une fixation des matériaux...

Pour être en conformité avec le français rationalisé, un texte ne doit pas comprendre de mots étrangers à ces trois familles. Il est important de souligner que les mots autorisés le sont dans une acception particulière, il est interdit de les employer dans une acception autre.

Les autres prescriptions du français rationalisé regroupent pêle-mêle des règles de niveaux différents :

- des règles lexico-syntaxiques, telles que l'interdiction de l'emploi de la forme passive, celle de certaines constructions à verbes support.
- des règles morphologiques interdisant la conjugaison des verbes à certains temps et modes grammaticaux.
- des règles stylistiques, imposant, par exemple, une limite à la longueur des phrases.
- des règles concernant l'organisation communicative de l'énoncé : elles imposent aux phrases d'être précises et rigoureuses, de ne véhiculer qu'une seule instruction.

Le lexique et les règles du français rationalisé sont regroupés dans un document appelé *guide du rédacteur*. Comme son nom l'indique, ce document est destiné aux rédacteurs techniques, il s'adresse à un public qualifié possédant une bonne connaissance de la langue. Les règles et le lexique regroupés au sein du guide du rédacteur ne peuvent être vus comme une grammaire générative du français rationalisé : il ne contiennent pas de règles générales de syntaxe, rien n'est dit des phénomènes d'accord, de la position relative des mots dans la phrase .... les règles d'une langue contrôlée ne décrivent que certains aspects de la langue contrôlée. Les relations qu'entretiennent les règles du français rationalisé, les règles du français courant et du français de l'aéronautique ne sont pas explicitées ([Lux & Dauphin, 1996]).

Une telle explicitation de l'articulation du français rationalisé, du français standard et du français de l'aéronautique nécessite une analyse linguistique fine tant des constructions attestées dans les corpus que des effets de l'application des règles du français rationalisé. On trouvera dans [Zajac & Heald, 1996] une analyse d'une règle particulière de l'anglais simplifié, imposant la présence d'un déterminant dans tout groupe nominal. L'étude montre qu'une telle règle ne se justifie pas toujours : elle peut dans certains cas modifier le sens d'une phrase. Une règle plus précise, permettant de traiter les seuls cas où l'introduction de l'article accroît la lisibilité de la phrase est donc souhaitable, mais elle se situera à un niveau de complexité bien supérieur à la règle actuelle. On peut s'attendre à ce que, de manière générale, la définition de règles plus fines que celles existant actuellement aboutisse à un système complexe de règles interdépendantes, auxquelles sont associées de nombreuses conditions d'application. Si un tel niveau de précision des règles d'une langue contrôlée est souhaitable et même nécessaire dans l'optique d'une mise en œuvre informatique, il n'est par contre pas adapté à une utilisation par un opérateur humain, qui était sa raison d'être première, comme l'indique clairement l'expression« guide du rédacteur». Les deux exigences sont difficile à concilier et il semble peu probable que l'on puisse proposer une formulation unique des règles du français rationalisé qui soit adaptée aussi bien à une utilisation humaine qu'à une interprétation informatique. La solution semble plutôt résider dans le développement d'outils informatiques aidant à la rédaction de textes en une langue contrôlée, déchargeant le rédacteur des aspects les plus ingrats que présente cette activité.

1.2.1 Langues contrôlées et traitement automatique du langage

Bien qu'élaborées indépendamment de préoccupations propres au traitement automatique de la langue (TAL), les langues contrôlées ont vite attiré l'attention des membres de cette communauté. Ces derniers, confrontés à la complexité et aux irrégularités de la langue dans son traitement informatique, ont vu dans cette « rationalisation » de la langue un espoir de traiter avec un objet plus facile à décrire et à manipuler. De plus, le rôle prépondérant que l'informatique a acquis dans le domaine de la documentation, et particulièrement de la documentation technique, a contribué au rapprochement des rédacteurs techniques et des informaticiens. Un curieux ménage à trois entre linguistes informaticiens, informaticiens de la documentation et rédacteurs techniques a vu le jour et l'on assiste aux débuts difficiles d'une cohabitation où se croisent des concepts aussi différents que balises SGML (Standard Generalised Markup Language), phénomènes de dépendances non bornées et clapets de sectionnement.

Comme nous l'avons observé en 1.2, les règles des langues contrôlées sont destinées à un public humain, elles mêlent des règles s'énonçant à des niveaux linguistiques très variés, allant de la morphologie à la pragmatique. Si certaines règles lexicales, morphologiques et syntaxiques peuvent être exploitées dans le cadre d'applications informatiques, d'autres, situées à des niveaux linguistiques moins bien formalisés, tels que la pragmatique et la sémantique, ne se laissent pas traiter facilement. Dans le domaine du TAL, deux grandes familles d'applications relatives aux langues contrôlées se sont dessinées : la vérification de la conformité de textes techniques à une langue contrôlée et la traduction automatique de textes d'une langue contrôlée vers une autre langue contrôlée.

La vérification de la conformité de textes techniques à une langue contrôlée est l'activité la plus importante dans ce domaine. Elle se situe souvent à la frontière de deux fonctionnalités, d'une part la détection d'erreurs en vue d'une correction manuelle par le rédacteur et d'autre part l'apprentissage de la langue contrôlée par le rédacteur. La plupart des projets de vérification prévoient en effet, lors de la détection d'une erreur, de proposer à l'attention du rédacteur la règle avec laquelle l'erreur se trouve en conflit, jouant ainsi un rôle didactique. Les projets de verification de la conformité de textes techniques sont nombreux. On citera à titre d'exemple le projet EGSC mené par le constructeur aéronautique Boeing [Wojcik & Holmback, 1996, Wojcik et al., 1990], le projet EUROCASTLE de l'Aérospatiale [Clémencin, 1996], le projet ClearCheck de Caterpillar [Hayes et al., 1996] et le projet SDD de Siemens [Schachtl, 1996]. Le succès de ce type d'applications provient en partie du fait que certaines entorses aux règles d'une langue contrôlée se laissent facilement détecter par une analyse sommaire des phrases, ce qui a permis d'aboutir assez rapidement à des outils opérationnels.

La traduction automatique de textes d'une langue contrôlée vers une autre bénéficie de deux avantages : d'une part, elle profite d'une expérience en traduction automatique riche de plus de quarante années et d'autre part, comme nous l'avions déjà noté pour les langues de spécialité, elle évite un écueil traditionnel de la traduction automatique, en limitant son intérêt à des textes appartenant à des domaines sémantiques restreints et répondant à des contraintes lexico-syntaxiques sévères. Après une période de méfiance des industriels à l'égard de la traduction automatique, datant de la publication du rapport de l'ALPAC en 1966 ([ALPAC, 1966]), l'intérêt pour cette discipline semble renaître dans les milieux industriels. En témoignent les projets menés par Caterpillar (AMT) [Hayes et al., 1996], General Motors (CASL) [Means & Godden, 1996] et Scania [Almqvist & Hein, 1996]. Le succès apparent du projet AMT qui doit aboutir à une version opérationnelle pour le couple anglais simplifié, français rationalisé dans le courant de l'année 1996 semble autoriser un certain optimisme quant à l'avenir de cette activité.

L'activité de reformulation, dont le but est de reformuler des textes techniques pour les rendre conformes à une langue contrôlée, apparaît comme le parent pauvre des applications informatiques. Elle peut être rapprochée des deux applications précédentes mais ne bénéficie pas des conditions qui sont, en partie, à l'origine de leur succès. Elle peut être vue comme de la traduction d'une langue non contrôlée vers une langue contrôlée mais, contrairement à la traduction entre langues contrôlées, elle ne peut faire d'hypothèses *a priori* sur la conformité des phrases à traduire à une langue contrôlée et doit, par conséquent, considérer un ensemble plus vaste de constructions lexico-syntaxiques que celles autorisées par la langue contrôlée. D'autre part, sa problématique n'est pas indépendante de celle de la vérification de conformité à une norme. Cette dernière peut en effet être vue comme une première étape de la reformulation, une phase de détection de structures non approuvées. Mais, là aussi, l'avantage de la vérification, à savoir la relative facilité à détecter des erreurs est mis à mal par la nécessité de les corriger. Nombreux sont les cas où une entorse à une règle est facile à détecter et difficile à corriger, on pensera notamment aux règles limitant la longueur des phrases.

Deux projets de reformulation de textes dans une langue contrôlée existent cependant. Le premier historiquement et le plus cité dans la littérature est le projet européen SECC (Simplified English Grammar and Style Checker/Corrector) [Adriaens, 1994]. Ce projet se base sur une langue contrôlée appelée COGRAM, élaborée à partir de plusieurs langues contrôlées existantes [Adriaens & Schreurs, 1992]. La tâche de reformulation est vue ici comme la traduction d'un texte en une langue contrôlée. Un système de traduction automatique existant a donc été adapté pour prendre en compte ce couple de langues inhabituel. Le système de traduction automatique utilisé est le système METAL, fruit de vingt ans de recherches ([Slocum, 1987, Bennett & Slocum, 1985]). Ce dernier est un système à transfert, dans lequel le composant de transfert est constitué d'un lexique mettant en relation des mots autorisés et des mots interdits qui leur sont synonymes et d'une grammaire mettant en relation structures interdites et structures autorisées. Dans le cas de SECC, la grammaire de transfert est composée de cent cinquante règles regroupées en quatre familles : des règles dites de contrôle textuel, de contrôle syntaxique, de contrôle lexical et de contrôle de ponctuation. Nous ne pouvons malheureusement pas donner plus de détails sur les règles car elles ne sont pas publiées. Les sorties du système consistent en un objet complexe, combinant la phrase de départ accompagnée de la liste des erreurs détectées et des règles permettant de les corriger. Lorsque les règles peuvent être appliquées automatiquement, elles le sont à la demande de l'utilisateur.

Le point qui mérite d'être souligné dans cette approche est le recours qui a été fait à un système existant de traduction automatique. L'emploi d'un tel système et son adaptation à l'activité de reformulation, décrit en partie dans [Thurmair, 1990], pose toutefois le problème du partage des ressources linguistiques nécessaires à l'analyse et à la génération dans le cadre de la reformulation. La proximité de la langue source et de la langue cible appelle en effet à un partage important des ressources linguistiques. La grammaire et le lexique d'une langue contrôlée et ceux de la langue dans laquelle elle s'inscrit présentent en effet une partie commune importante qu'il est peu satisfaisant de dupliquer. Ce partage n'est pas facile à effectuer dans un système conçu initialement pour la traduction d'une langue vers une autre. Dans le cas de SECC, l'analyse et la génération ne partagent pas la même grammaire, elles ont par contre un lexique commun. La distinction entre entrées lexicales conformes et entrées lexicales non conformes est établie au niveau du lexique de transfert : à toute entrée lexicale non approuvée par la langue contrôlée est associée une règle permettant de la remplacer par un synonyme approuvé. Un plus grand partage des ressources linguistiques nous semble souhaitable dans un système de reformulation.

Nous ne parlerons pas ici du second projet de reformulation développé par la firme Cap Volmac [van der Eijck et al., 1996] dont les détails, tant du point de vue linguistique qu'informatique, ne sont pas publiés.

L'application de reformulation de textes techniques dans une langue contrôlée nous a semblé intéressante à trois titres.

- La rédaction de textes dans une langue contrôlée semble être un exercice fastidieux, souvent ressenti comme une contrainte par le rédacteur, comme le note [Goyvaerts, 1996]. L'activité de rédaction est en elle-même un exercice difficile, que la prise en compte d'une norme ne fait que compliquer. L'application de reformulation automatique nous apparaît comme une aide utile dans l'activité de rédaction, qui pourra peut-être, à terme, libérer le rédacteur de contraintes souvent ingrates.

- En ce qui concerne le TAL, la reformulation nous apparaît comme un axe de recherche riche, il se situe à la limite de la correction et de la traduction automatique et fait appel à des activités différentes du TAL, telles que l'analyse et la génération.

- D'un point de vue plus linguistique, la mise en œuvre automatique de la reformulation dans une langue contrôlée, qui tente de codifier un certain style, pose le problème du rapport complexe qu'entretiennent la stylistique et d'autres niveaux linguistiques que sont la morphologie, la syntaxe, la sémantique et la pragmatique. L'activité de reformulation automatique nécessite une « traduction » de règles stylistiques en termes lexico-syntaxiques, permettant, ce faisant, d'expliciter les liens complexes qui existent entre ces niveaux, à l'image des grammaires stylistiques de [DiMarco & Hirst, 1988].

1.3 Théorie Sens-Texte et reformulation

La mise en œuvre informatique de la reformulation est confrontée à deux problèmes distincts. D'une part, et comme nous l'avons déjà dit, les règles de la langue contrôlée doivent être formulées de façon à être interprétées par un ordinateur. D'autre part, nous avons vu qu'une langue contrôlée, contrairement à un langage informatique, n'est pas définie par une grammaire, mais par quelques règles. Le traitement automatique de textes dans une langue contrôlée, que ce soit l'analyse, la génération ou encore la paraphrase nécessite par conséquent, en sus des règles propres à la langue contrôlée, de nombreuses connaissances linguistiques générales indépendantes de cette dernière. Une solution pourrait consister à « compléter » les règles d'une langue contrôlée avec des règles de syntaxe générales de façon à constituer une grammaire exhaustive de la langue contrôlée, à l'image des projets de description d'une langue de spécialité évoqués en 1.1. Nous avons préféré intégrer les règles d'une langue contrôlée à une théorie linguistique existante, tout en distinguant ce qui relève de la langue contrôlée et ce qui concerne la connaissance linguistique générale, limitant notre apport au premier membre de cette dichotomie. De ce point de vue, plus la théorie linguistique sera riche, prenant à sa charge un grand nombre d'aspects linguistiques, plus la dichotomie pourra être respectée et notre apport se limitera à des aspects propres aux langues contrôlées.

La Théorie Sens-Texte (*TST*), qui sera décrite au chapitre 2, offre de ce point de vue des avantages considérables grâce notamment à son système de paraphrase. L'application de reformulation de textes techniques dans une langue contrôlée peut être vue comme une application contrôlée du système de paraphrase de la *TST*. Le partage des rôles entre la *TST* et la langue contrôlée se fait naturellement. La *TST* offre un cadre linguistique pour la paraphrase tandis que les contraintes propres à la langue contrôlée permettent de guider les choix paraphrastiques. Il va sans dire que seules certaines règles d'une langue contrôlée comme le français rationalisé dans son état actuel peuvent être exprimées par des règles de paraphrase de la *TST*. D'autres règles pourront être « traduites » en règles de paraphrase de la *TST*. Cette dernière joue ici un rôle méthodologique en proposant une base de règles élémentaires de paraphrase dans laquelle exprimer des règles d'une langue contrôlée. Cette base pourra s'avérer trop pauvre, elle devra alors être enrichie de nouvelles règles.

La variété et l'étendue des problèmes abordés dans ce chapitre ont permis de situer le contexte de notre recherche. Elles ont aussi mis en évidence l'importance de faire des choix pour dégager une problématique de thèse conforme aux deux points que nous avons évoqué dans la présentation. La problématique choisie relève du TAL et de la *TST*. Elle constitue une tentative de mise en œuvre automatique du système de paraphrase complet de la *TST* prenant en entrée une phrase P pour donner en sortie une paraphrase de P répondant à un certain nombre de contraintes.

L'originalité de notre recherche par rapport à la *TST* nous semble résider dans les deux points suivants :

- La *TST*, tout en se présentant comme un système bi-directionnel, a toujours privilégié la transition Sens $\longrightarrow$Texte (génération) à la transition Texte $\longrightarrow$Sens (analyse). Nous proposerons un mode de représentation des connaissances linguistiques adapté à la double transition Sens $\longleftrightarrow$ Texte.

- D'autre part, le système de paraphrase de la *TST* permet théoriquement de générer un grand nombre de paraphrases d'une phrase donnée. L'application de reformulation de textes techniques dans une langue contrôlée nous semble représenter une application originale de ce système.

Du point de vue du TAL l'originalité de notre recherche nous semble également double :

- Proposer un formalisme pour grammaires de dépendances pour l'analyse et la reformulation, intégrant des techniques récentes nées dans le monde des grammaires syntagmatiques.

- Envisager la tâche d'analyse syntaxique dans le cadre des grammaires de dépendances. L'analyse syntaxique a connu dans le cadre des grammaires syntagmatiques, avec notamment le développement des langages informatiques, un développement extraordinaire. Les grammaires de dépendances qui n'ont pas franchi le domaine de la langue naturelle n'ont pas connu un tel développement. Nous espérons que cette thèse contribuera au développement de cette discipline.

Les choix que nous avons dû effectuer pour dégager une problématique de thèse nous ont éloigné de plusieurs aspects abordés dans ce chapitre, principalement du monde des langues contrôlées qui sont paradoxalement présentées comme le point de départ de notre recherche.

1.4 Organisation de la thèse

Notre travail s'inscrit dans le cadre de la *TST*, c'est pourquoi nous commencerons par une description de cette théorie dans le chapitre 2. Ce dernier débutera sur une présentation des différents aspects de la *TST* concernant la paraphrase. Nous commencerons par essayer de cerner les limites de la notion de paraphrase dans la *TST*. Nous décrirons ensuite les différents éléments de la *TST* importants pour la mise en œuvre de la paraphrase : les niveaux de représentation, le dictionnaire explicatif combinatoire et les règles de paraphrase. La description que nous proposons de la théorie est neutre, elle est indépendante de notre application. C'est dans la seconde partie du chapitre que nous introduirons la problématique de la reformulation, en abordant les problèmes de la profondeur et du contrôle de la reformlation. A l'issue de ce chapitre, le principe général de reformulation au sein de la *TST* sera proposé. Le reste de la thèse consistera principalement à étudier la mise en œuvre de ce principe en s'intéressant d'une part à la représentation des connaissances nécessaires à la reformulation et d'autre part aux aspects procéduraux de la mise en œuvre.

Le chapitre 3 va poser les bases du modèle de reformulation en définissant les arbres élémentaires qui sont des structures lexico-syntaxiques complexes, les relations pouvant être définies entre eux, les règles permettant d'établir ces relation ainsi que des opérations de manipulation des arbres élémentaires. Ces derniers seront présentés comme le résultat d'une réorganisation des composants de la *TST* motivée par des contraintes de mise en œuvre informatique ainsi que par les particularités de la problématique de la reformulation. La définition des arbres élémentaires, des règles et des opérations de manipulation vont permettre de représenter les règles d'écriture d'une langue contrôlée et les connaissances linguistiques nécessaires à la mise en œuvre de la reformulation dans un même formalisme. Le processus complet de reformulation pourra alors être proposé.

Deux notions importantes de notre approche, les arbres élémentaires de surface et l'opération d'attachement permettant de les combiner seront présentés en détail dans le chapitre 4. Nous verrons notamment quelles connaissances sont représentées dans les arbres élémentaires et la manière dont elles sont prises en compte dans l'opération d'attachement. Cette présentation détaillée de ces deux notions va aussi être l'occasion de comparer ces deux parties de notre formalisme à leurs équivalents dans d'autres formalismes grammaticaux de la littérature.

Les aspects procéduraux de notre travail seront présentés dans les deux chapitres 5 et 6. Nous décrirons dans le premier la phase d'analyse syntaxique de la phrase à reformuler qui se décompose en deux étapes, une étape de construction de la structure syntaxique de surface suivie d'une étape de construction d'une structure syntaxique profonde. Cette dernière subira des transformations visant à la mise en conformité avec une langue contrôlée lors de la phase de transformation qui sera décrite dans le chapitre 6. C'est aussi dans ce chapitre que nous décrirons la phase de regénération d'une phrase à partir de la structure syntaxique profonde transformée, dernière étape du processus de reformulation.

Différentes difficultés rencontrées dans les deux derniers chapitres, pour la mise en œuvre du processus de reformulation vont nous amener à modifier et enrichir certains aspects du formalisme proposé au chapitre 3. Nous ferons le point sur les modifications apportées et sur le « nouveau » formalisme qui en résulte dans l'annexe A. Dans l'annexe B nous présenterons la grammaire du français et les règles d'écriture que nous avons implémenté dans notre formalisme. Finalement, nous proposerons dans l'annexe C des éléments d'implémentation de l'algorithme d'analyse décrit dans le chapitre 5, qui constitue l'étape la plus complexe du processus de reformulation.

Chapitre 2

Paraphrase et reformulation dans la Théorie Sens-Texte

Nous avons évoqué dans le chapitre précédent notre volonté de bâtir un système de reformulation sur une théorie linguistique, qui est, dans notre cas, la Théorie Sens-Texte. Cette démarche nous permet de nous reposer sur cette théorie et d'adopter un certain nombre de ses hypothèses, telles que les niveaux de représentation des énoncés qu'elle postule et son système de paraphrase. La place importante qu'occupe la *TST* dans notre travail nous semble justifier une présentation détaillée de cette dernière en 2.1. Nous commencerons par présenter les différents niveaux de représentation d'un énoncé proposés par la *TST*, nous décrirons ensuite le *Dictionnaire Explicatif Combinatoire* qui occupe une place importante dans l'organisation de la *TST* et, finalement, le système de paraphrase qui constitue le cœur du processus de paraphrase. Ces différents éléments vont nous permettre de décrire en détail les mécanismes de la paraphrase dans la *TST*, d'identifier les différents points où des choix paraphrastiques sont possibles ainsi que les problèmes auxquels ils se heurtent. Nous étudierons ensuite, en 2.2, deux scénarios possibles de mise en œuvre de la reformulation dans le cadre de la *TST*, pour en choisir un que l'on rapprochera d'une architecture à transfert dans le cadre de la traduction automatique. Finalement, en 2.2.3, nous présenterons deux problèmes théoriques auxquels est confronté le processus de reformulation dans le cadre de la *TST*, indépendament de la manière dont nous l'avons mis en œuvre dans cette thèse. A l'issue de ce chapitre, les grandes lignes du processus de reformulation seront tracées : nous aurons identifié les trois étapes de la reformulation que sont l'analyse, la transformation de la structure syntaxique profonde et la regénération.

2.1 Sens et paraphrase dans la *TST*

La paraphrase occupe au sein de la *TST* une position de choix. La« thèse centrale» de la démarche de ses auteurs, telle qu'elle est formulée dans [Mel'čuk, 1988b], propose en effet que « *une des tâches primordiales de la linguistique théorique contemporaine est l'élaboration d'une théorie de la paraphrase langagière*». La *TST* est présentée selon ses auteurs comme une modélisation de la compétence paraphrastique des locuteurs d'une langue donnée. Plus concrètement, la *TST* propose de décrire les correspondances entre, d'une part, un sens et les différents textes permettant de communiquer ce sens et, dans une moindre mesure, un texte et les différents sens que ce texte véhicule.

On s'intéressera particulièrement dans cette section à la transition Sens ⟶Texte et donc aux aspects *paraphrastiques* de la théorie. Cet aspect a été largement privilégié, au détriment de la direction Texte ⟶Sens, aussi bien dans les travaux tournant autour de la *TST* que dans la théorie elle-même. A tel point que l'organisation générale de la *TST* porte les marques de ce choix, comme nous le verrons lorsque nous analyserons certains aspects de la transition Texte ⟶Sens en 3.2.

L'étude et la modélisation de la double transition Sens ⟷Texte appelle des précisions quant aux notions de texte et de sens. Comme le note [Polguère, 1990], l'usage qui est fait ici du terme *texte* est assez trompeur, il laisse croire que la théorie cherche à établir une correspondance entre un sens et un véritable texte, composé de phrases et possédant une organisation interne cohérente. Il faut en réalité entendre par texte, dans la *TST*, la réalisation phonétique d'un énoncé de la taille d'une phrase ou d'une partie de phrase. Dans le cadre de notre étude, nous ne nous intéresserons pas à l'aspect phonétique de la transition Sens ⟶Texte, nous nous limiterons à la réalisation dite *morphologique de surface* (*RMorphS*) de l'énoncé : une chaîne linéaire de mots formant une phrase ou une partie de phrase.

Si la notion de texte dans la *TST* est facile à appréhender, la notion de sens nécessite, par contre, un développement plus important. Le sens est défini dans la *TST* comme « *l'invariant des paraphrases langagières ou la seule propriété commune à tous les énoncés ayant le même sens*», il est matérialisé par une *Représentation Sémantique* (*RSém*). Cette définition renvoie à la notion d'identité de sens qui est considérée comme primitive dans la *TST*. On n'essayera pas de savoir dans quelles conditions deux énoncés ont même sens, ce travail sera laissé à l'intuition des locuteurs de la langue. Une telle attitude suppose néanmoins l'existence d'un consensus entre locuteurs sur la synonymie de plusieurs énoncés. Mais, comme le fait remarquer [Fuchs, 1982], un tel consensus est difficile à trouver car plusieurs personnes interrogées ne s'accordent souvent pas sur le point de savoir si deux séquences X et Y « signifient la même chose ». Derrière cette absence de consensus entre locuteurs se pose le problème de la possibilité même d'une identité sémantique d'énoncés différents. La pratique montre en effet qu'une analyse poussée permet généralement de mettre à jour des différences sémantiques entre n'importe quelles expressions linguistiques. L'identité sémantique ne serait-elle en définitive que le résultat d'un manque de finesse de l'analyse sémantique ? Ou, comme le dit C. Fuchs : « *est il vraiment possible de dire exactement la même chose de plusieurs façons ?*».

Analysant l'attitude adoptée par différents courants linguistiques face à ce problème, C. Fuchs identifie deux grandes tendances : « *un désinterêt à l'égard de la paraphrase (il n'y aurait, en définitive, pas de relation paraphrastique en langue, puisqu'à chaque fois que l'on croit en tenir une, il est possible de montrer qu'en réalité il existe des différences sémantiques)* », ou bien, au contraire, « *un regain d'interêt pour la paraphrase (nécessité d'abandonner cette idée « peu subtile » de l'identité de sens, et de pousser plus avant l'analyse de la paraphrase en termes de ressemblance, de proximité, d'équivalence sémantique)* ».

C'est la seconde attitude qui a été adoptée par les auteurs de la *TST*, qui proposent un point de vue plus nuancé sur la paraphrase et la notion de sens d'un énoncé. Ce point de vue s'appuie sur les trois points suivants :

Une attitude « permissive » vis-à-vis de l'identité de sens entre plusieurs énoncés

Si les exemples d'énoncés sémantiquement identiques sont rares, il n'en reste pas moins vrai que certaines distinctions sémantiques qui existent dans la langue sont ignorées dans le discours. I. Mel'čuk tente d'expliquer ce phénomène par la notion de *neutralisation sémantique contextuelle*, selon laquelle certaines différences sémantiques sont neutralisées dans un contexte d'énonciation donné. Deux unités lexicales qui se distinguent sémantiquement peuvent perdre leurs différences dans un certain contexte. A titre d'exemple, [Mel'čuk, 1988b] cite les verbes *s'arrêter* et *cesser* qui s'opposent dans les contextes suivants : *Ses visites se sont arrêtées durant le mois de mai vs Ses visites ont cessé durant le mois de mai* peuvent être considérées comme équivalentes dans les expressions : *La pluie s'est arrêtée vs La pluie a cessé.*

Les auteurs de la *TST* voudraient que se reflète dans leur modèle une certaine « *nonchalance* » de l'analyse sémantique, consistant à ne pas rechercher dans l'interprétation sémantique de deux énoncés des distinctions qui ne sont pas nécessairement observées dans la réalité langagière. Il est peu habituel de parler de « *nonchalance* » dans le cadre d'une modélisation formelle et on pourra distinguer deux moyens radicalement différents de la mettre en œuvre dans un contexte de production d'énoncés paraphrastiques :

- Un manque de précision dans la représentation des énoncés et des mécanismes de génération de paraphrases. Dans une telle perspective, il n'est pas possible de représenter certaines nuances sémantiques et donc de les respecter au niveau de la paraphrase. La nonchalance, dans un pareil cas, est induite par la pauvreté du modèle ; on pourra alors parler de nonchalance incontrôlée.

- Une représentation extrêmement poussée des nuances sémantiques, accompagnée d'un moyen de les ignorer en fonction d'un contexte particulier ; on parlera de nonchalance simulée. C'est, on le conçoit, une attitude préférable à la précédente dans la mesure où elle permet de simuler une certaine nonchalance tout en contrôlant les glissements de sens éventuels.

On verra que dans la *TST*, la nonchalance est quelquefois simulée et d'autre fois incontrôlée, engendrant des problèmes d'exactitude des paraphrases qu'elle permet de générer (voir 2.2.3).

Identification d'une paraphrase langagière

Une distinction est effectuée entre *paraphrases langagières* et *paraphrases non langagières*, distinction que l'on retrouve chez C. Fuchs entre *paraphrases linguistiques* et *paraphrases référentielles.* Deux paraphrases seront dites langagières lorsque la synonymie peut être établie sans qu'il soit fait usage de connaissances extralinguistiques. Les deux exemples suivants, empruntés à [Mel'čuk & Zholkovsky, 1970], illustrent des paraphrases non langagières.

Le peuple allemand luttait contre le fascisme.
Des Alpes à la Mer du Nord, du Rhin à l'Oder, le peuple continuait la guerre d'Edgar André.

Plus de la moitié du territoire de l'Egypte est occupé par des déserts.
En Egypte, les déserts occupent un territoire plus vaste que la France.

Pour décider si de telles phrases sont synonymes, il convient de connaître la géographie de la France et de l'Allemagne et de savoir qui était Edgar André. Ce type de paraphrases, faisant appel à des connaissances encyclopédiques, ne sera pas considéré dans le cadre de la *TST*. Le sens vu comme invariant de la paraphrase langagière (le *sens langagier*) ne prend pas en charge les connaissances extralinguistiques permettant de juger comme synonymes les phrases des exemples précédents. Ces dernières présenteront, par conséquent, des *RSém* différentes. La distinction opérée entre paraphrase linguistique et paraphrase extralinguistique n'est pas toujours facile à établir, si certaines paraphrases sont clairement linguistiques et d'autres extralinguistiques, il subsiste néanmoins tout un ensemble de paraphrases qu'il est difficile de classer dans un des deux membres de cette opposition. Plutôt qu'une distinction stricte entre paraphrases linguistiques et extralinguistiques c'est en terme d'un continuum que doit être pensé le caractère linguistique d'une paraphrase.

Décomposition du sens selon trois « dimensions »

Le sens langagier est composé dans la *TST* d'éléments de trois types :

1. Le sens situationnel, qui représente l'état de chose dont il s'agit : les objets, les événements ... ainsi que les relations qu'ils entretiennent entre eux.

2. Le sens communicatif qui représente l'organisation du message par le locuteur.

3. Le sens rhétorique qui représente les effets expressifs ou artistiques visés par le locuteur[1].

[1] La frontière entre le sens communicatif et le sens rhétorique n'est pas toujours facile à tracer et on pourra regretter qu'elle n'ait pas été mieux précisée dans la *TST*.

Pour être synonymes, dans le cadre de la *TST*, deux phrases devront partager la première composante du sens, le sens situationnel. C'est en quelque sorte la condition minimale que doivent vérifier deux phrases en relation paraphrastique. Cette définition est arbitraire : elle aurait pu exiger en sus du partage de la composante situationnelle, celui des composantes communicatives et/ou rhétoriques. Le point important est d'avoir dissocié ces trois composantes du sens linguistique. La définition de la paraphrase peut sembler à côté secondaire. On peut parler dans ce cas de nonchalance simulée, dans la mesure où certaines distinctions représentées dans le modèle sont ignorées dans la paraphrase.

2.1.1 Les niveaux de représentation

Le sens langagier d'un énoncé phrastique étant représenté dans la *TST* sous la forme d'une *Représentation Sémantique* (*RSém*), on trouvera aux deux extrémités de la correspondance Sens $\longleftrightarrow$Texte, une ou plusieurs *RSém* et une ou plusieurs *RMorphS* (Représentation Morphologique de Surface).

La grande complexité de la correspondance *RSém*$\longleftrightarrow$*RMorphS* a poussé les auteurs à postuler trois niveaux de représentation intermédiaires, un niveau syntaxique profond, un niveau syntaxique de surface et un niveau morphologique profond. A chacun de ces niveaux correspond un langage formel permettant de représenter un énoncé. Nous allons décrire successivement ces niveaux dans les paragraphes suivants, en mettant l'accent sur les moyens de représentation que chaque niveau fournit et sur la façon dont une même information est représentée différement à des niveaux adjacents.

La représentation sémantique

Au niveau sémantique de la *TST* est défini un langage permettant de décrire sous la forme d'une *RSém* le sens d'un ensemble de paraphrases, dans l'acception particulière de la paraphrase que nous avons évoqué ci-dessus. Les trois composantes du sens se retrouvent ici dans la décomposition de la *RSém* en trois structures : une structure sémantique (*SSém*), une structure communicative et une structure rhétorique.

La figure 2.1 présente une *SSém* qui peut se lire en français de la façon suivante : *le mécanicien s'assure du bon fonctionnement de la pompe électrique qui est contenue dans le circuit.*

Nous commencerons par décrire la structure sémantique qui, outre son rôle majeur au sein de la *RSém*, est la plus aboutie des trois composantes du niveau sémantique. Une structure sémantique est représentée par un graphe connexe. Les nœuds de ce graphe sont étiquetés par des unités sémantiques ou *sémantèmes* qui correspondent au sens d'un lexème[2] ou d'un phrasème[3]. Cette

[2]Un lexème est un mot pris dans une acception déterminée.

[3]Un phrasème est une expression multilexémique ayant des propriétés imprévisibles à partir des propriétés de

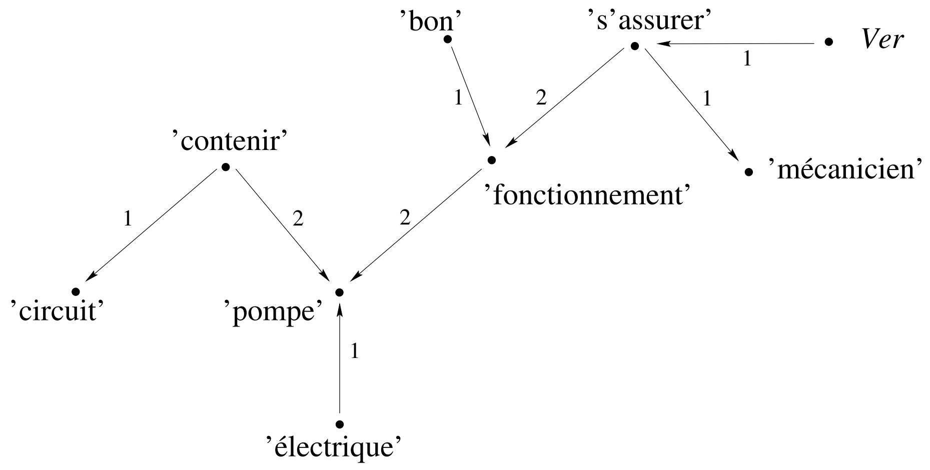


FIG. 2.1: Une structure sémantique

correspondance entre sémantème et lexème illustre le caractère linguistique du niveau sémantique de la *RSém.* Les unités de sens correspondent ici à des mots. Il n'y a donc pas, au sein de la *TST*, de niveau de représentation indépendant de la langue.

Les sémantèmes se répartissent selon deux grandes classes :

- Les foncteurs (dans notre exemple *contenir, s'assurer, bon, électrique et Ver*[4]) qui se subdivisent eux-mêmes en trois sous-classes : les prédicats, les quantificateurs et les connecteurs logiques.

- Les noms d'objets ou de classes d'objets (*circuit, pompe, mécanicien*).

On notera dans notre exemple l'aspect arbitraire de la « finesse » des sémantèmes. Le foncteur *contenir*, par exemple, aurait pu être décomposé en un sous-réseau constitué de sémantèmes plus primitifs. C'est là un point important qui sera repris en 2.1.3.

Les arcs de la structure sémantique sont étiquetés par un numéro qui n'a aucune signification en soi mais qui sert à identifier les arguments d'un foncteur. Le rôle de chaque argument par rapport à son foncteur est fixé dans l'article correspondant au foncteur, dans le lexique. Les arcs ont pour origine un foncteur et pour extrémité un objet ou un autre foncteur. Il convient de bien distinguer le caractère purement distinctif de l'étiquetage des dépendances sémantiques de la *TST*, des étiquettes des arcs des réseaux sémantiques que l'on retrouve en intelligence artificielle ou dans certaines approches linguistiques. De plus, l'étiquetage est arbitraire : deux arguments de deux foncteurs différents, ayant même numéro n'entretiendront pas le même rapport avec leurs foncteurs respectifs. L'argument numéro 1 du foncteur *manger*, par exemple, n'entretiendra pas

ses constituants.

[4] *Ver* est une fonction lexicale (concept décrit ci dessous) elle correspond au sens 'tel qui doit être', 'correct'

nécessairement le même rapport avec son foncteur que celui qu'entretiennent le foncteur *boire* et son argument numéro 1.

Se greffent sur la structure sémantique les deux autres composants de la *RSém* :

- La structure rhétorique, qui spécifie le style et les caractéristiques rhétoriques que le locuteur veut conférer à son énoncé. C'est une composante de la *RSyntP* qui n'est pas très développée dans la *TST*. Elle est marquée dans la *RSém* de la figure 2.2 de façon extrêmement sommaire par le sigle PND (Prosodie Neutre Déclarative).

- La structure communicative qui permet de représenter ce que le locuteur prend comme point de départ *vs* ce qu'il prend comme point d'arrivée ; ce qu'il veut poser d'abord *vs* ce qu'il veut discuter ensuite ; ce qu'il présuppose *vs* ce qu'il affirme ; ce qu'il veut renvoyer à l'arrière-plan. Elle est représentée dans la *TST* par une décomposition de la structure sémantique en thème et rhème, décomposition matérialisée par des zones identifiées à l'aide des lettres T et R dans l'exemple de la figure 2.2.

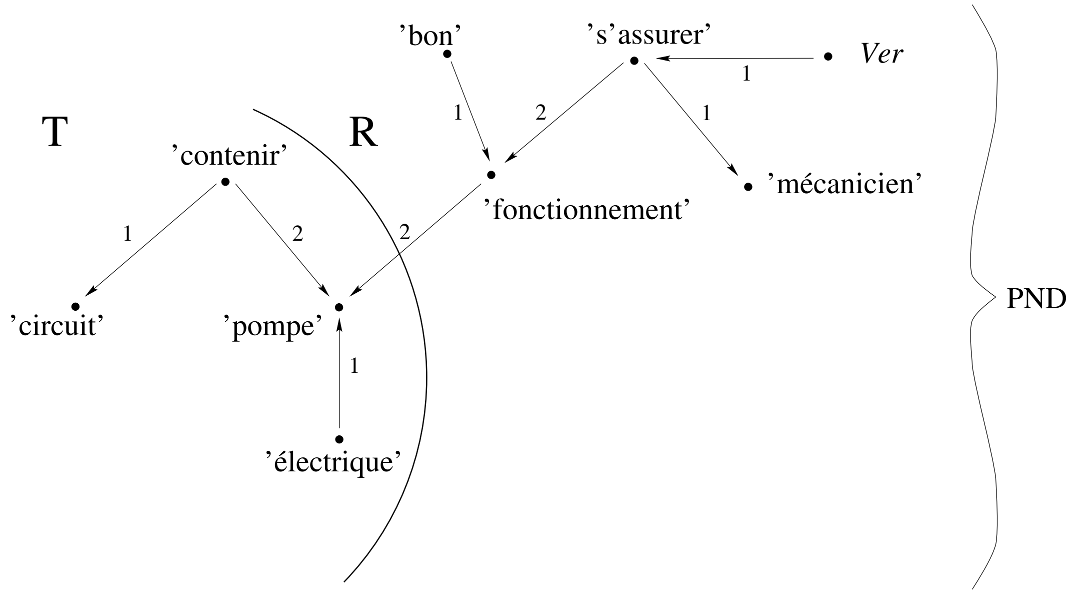


Fig. 2.2: Une représentation sémantique

La représentation syntaxique profonde

On retrouve dans la *RSyntP* la décomposition en une structure principale (ici la structure syntaxique profonde) et des structures secondaires déjà présentes dans la *RSém*. Deux *RSyntP*

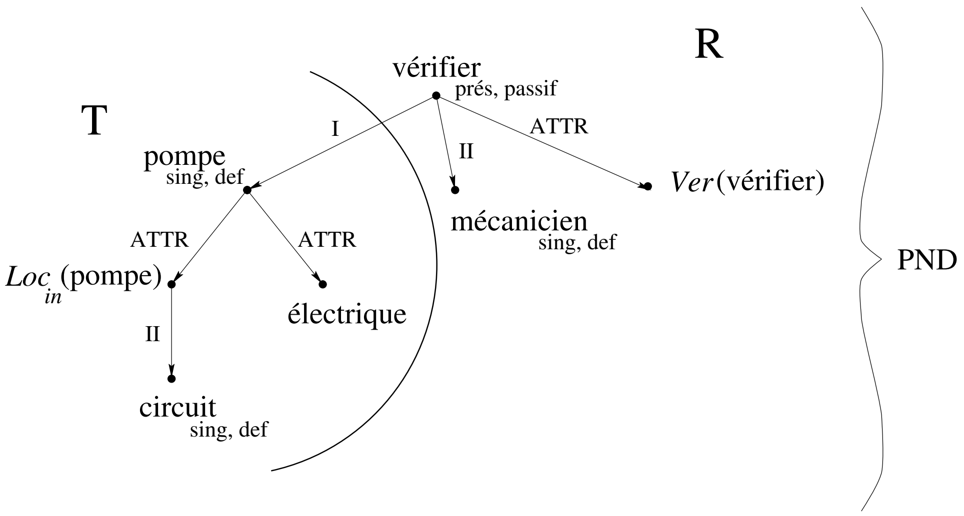


Fig. 2.3: Une *RSyntP* correspondant à la phrase *la pompe électrique du circuit est vérifiée soigneusement par le mécanicien*

correspondant à la *RSém* de la figure 2.2 sont données en 2.3 et 2.4.

La structure syntaxique profonde se présente sous la forme d'un arbre de dépendances non ordonné : la distribution spatiale de ses nœuds n'est pas pertinente. Ses branches sont étiquetées de noms de *relations syntaxiques profondes* qui sont des relations syntaxiques universelles. Neuf relations sont distinguées : six relations actancielles (I, II, III...), une relation modificatrice (ATTR), correspondant à des épithètes, compléments de noms ..., une relation coordinative (COORD) et une relation représentant des constructions extrastructurales, telles que les constructions parenthétiques (APPEND). Chaque relation représente une abstraction de plusieurs relations syntaxiques. La relation I, par exemple, représente aussi bien la relation liant un verbe à son sujet ($fonctionne \xrightarrow{I} pompe$) qu'un nom à l'un de ses compléments ($fonctionnement \xrightarrow{I} pompe$).

Les nœuds de la structure syntaxique profonde sont étiquetés par des *lexèmes profonds* correspondant aux mots de la phrase. Certains mots de la phrase n'apparaissent pas tels quels dans la structure syntaxique profonde :

- Les lexèmes« vides», moyens d'expression syntaxique, ne sont pas représentés. On trouvera, entre autres, dans cette catégorie, les auxiliaires, les prépositions régies.
- Les phrasèmes (expressions multilexémiques dont les propriétés ne peuvent pas être déduites des propriétés de leurs composants) sont représentés par un seul nœud.
- Les pronoms et mots pronominaux ne sont pas représentés, seuls leurs antécédents apparaissent.

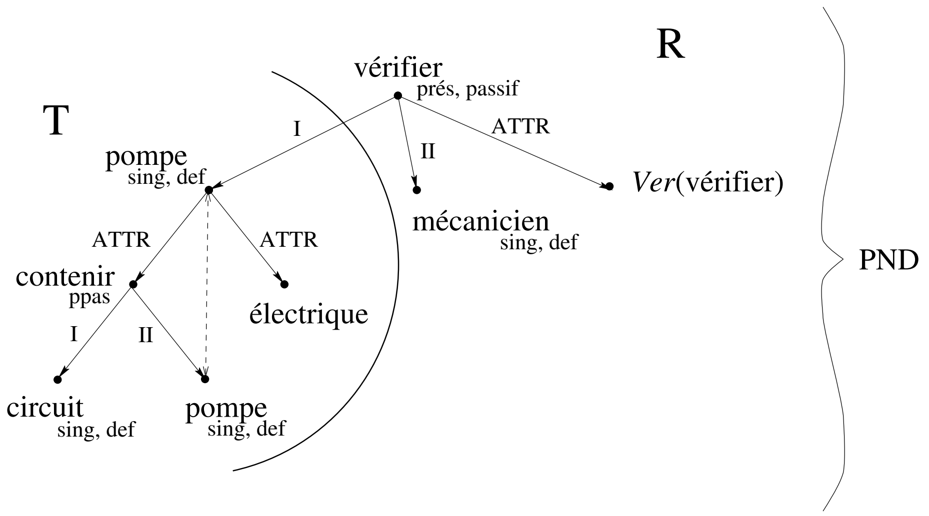


FIG. 2.4: Une *RSyntP* correspondant à la phrase *La pompe électrique contenue dans le circuit est vérifiée soigneusement par le mécanicien*

- Certains lexèmes choisis en fonction d'autres lexèmes seront représentés par le biais de *fonctions lexicales*. Une fonction lexicale[5] permet de formaliser, sous la forme d'une fonction[6], une relation sémantique entre deux lexèmes L1 et L2 ($F(L1) = L2$) qui est réalisée dans la langue d'une façon non prévisible. Autrement dit, le choix lexical pour exprimer un sens donné (représenté par la fonction lexicale) dans le contexte d'un lexème L1 (le mot clé) n'est pas libre : il est restreint aux valeurs de $F(L1)$. Les relations sémantiques représentées par des fonctions lexicales sont très variées. La fonction $Magn$, par exemple, exprime la notion d'intensité ($Magn(prix)$ = *haut*, *élevé* $Magn(malade)$ = *très*, *gravement*). Deux fonctions lexicales apparaissent dans la *RSyntP* de la figure 2.3 : la fonction Ver qui apparaissait déjà sous la forme d'un foncteur dans la *RSém* et la fonction Loc_{in} qui représente une préposition signifiant « se trouvant dans ».

Lors de la transition *RSém* ⟶*RSyntP*, certains sous-réseaux sémantiques seront remplacés par des lexèmes dont le sens correspond au sous-réseau. Cette opération de reconnaissance de sous-réseaux et de remplacement de sous-réseaux sémantiques par des lexèmes « complexes » est effectuée par l'opération de *lexicalisation*. Dans notre exemple, le réseau sémantique *s'assurer du bon fonctionnement* a été remplacé par le lexème *vérifier*. On voit se dessiner dans la structure syntaxique profonde la structure lexico-syntaxique de la phrase : les réseaux de sémantèmes de la représentation sémantique correspondant à des lexèmes de la langue ont été remplacés par ces derniers et la structure de graphe du niveau sémantique a laissé la place à une structure arborescente. Certains traits morphologiques sont représentés au niveau des lexèmes profonds : il s'agit

[5] La définition des fonctions lexicales proposée ici est empruntée à [Alonso Ramos & Tutin, 1993].

[6] Il ne s'agit pas à proprement parler d'une fonction dans la mesure où elle peut faire correspondre à un lexème plusieurs autres.

de traits morphologiques sémantiquement pleins, tels que le temps et le mode des verbes, et le nombre des mots. Les autres traits morphologiques sont des moyens d'expression syntaxique, ils seront représentés dans la *RMorphP*.

Se greffent à la structure syntaxique profonde, pour former la *RSyntP*, trois structures secondaires :

- La structure anaphorique profonde. Elle est composée de flèches bidirectionnelles représentant la coréférence de deux lexèmes dans la structure syntaxique profonde (voir figure 2.4). L'interprétation de cette structure donnera naissance au niveau syntaxique de surface à des pronoms ou à d'autres phénomènes anaphoriques.

- La structure communicative profonde. Elle consiste, à l'instar de son équivalent au niveau sémantique, en un découpage de la structure syntaxique profonde selon un thème et un rhème. On pourra remarquer que la structure communicative de la représentation sémantique a déjà été prise en compte lors de la transition *RSém* $\longrightarrow$*RSyntP*. Dans notre exemple, c'est du fait du découpage thème/rhème de la *RSém* que le sous-graphe sémantique correspondant à *s'assurer du bon fonctionnement* été remplacé par le lexème *vérifier* à la diathèse passive, permettant ainsi de placer le thème (*pompe*) en début de phrase. Une telle organisation linéaire de la phrase ne pouvait être réalisée avec le verbe *s'assurer*.

- La structure prosodique profonde Elle représente les aspects de la prosodie sémantiquement pertinents, tels que les aspects prosodiques indiquant le caractère interrogatif ou affirmatif d'une phrase. Les aspects de la prosodie découlant de considérations syntaxiques ne sont pas représentés à ce niveau.

On remarquera que la structure rhétorique apparaissant au niveau sémantique a maintenant été intégrée à la structure syntaxique profonde ; elle a été interprétée lors de la transition *RSém* $\longrightarrow$ *RSyntP* en termes de choix lexicaux et de structures syntaxiques.

La *RSyntP* de la figure 2.4 diffère de celle de la figure 2.3 par la lexicalisation du sémantème *'contenir'*, préposition Loc_{in} dans un cas et verbe *contenir* dans l'autre. Dans ce dernier cas, le nœud *pompe*, qui occupe deux fonctions syntaxiques différentes, est dupliqué et une relation de coréférence est établie entre les deux nœuds.

La représentation syntaxique de surface

La représentation syntaxique de surface est constituée de quatre sous-structures. Les structures prosodiques, communicatives et anaphoriques sont pratiquement similaires à leurs correspondants respectifs en syntaxe profonde. Seule la structure syntaxique de surface diffère sensiblement de la structure syntaxique profonde. Elle se présente toujours sous la forme d'un arbre de dépendance mais sa configuration ainsi que la nature de ses nœuds et de ses branches la distingue d'un arbre syntaxique profond.

Une *RSyntS* correspondant à la *RSyntP* de la figure 2.3 a été représentée en 2.5.

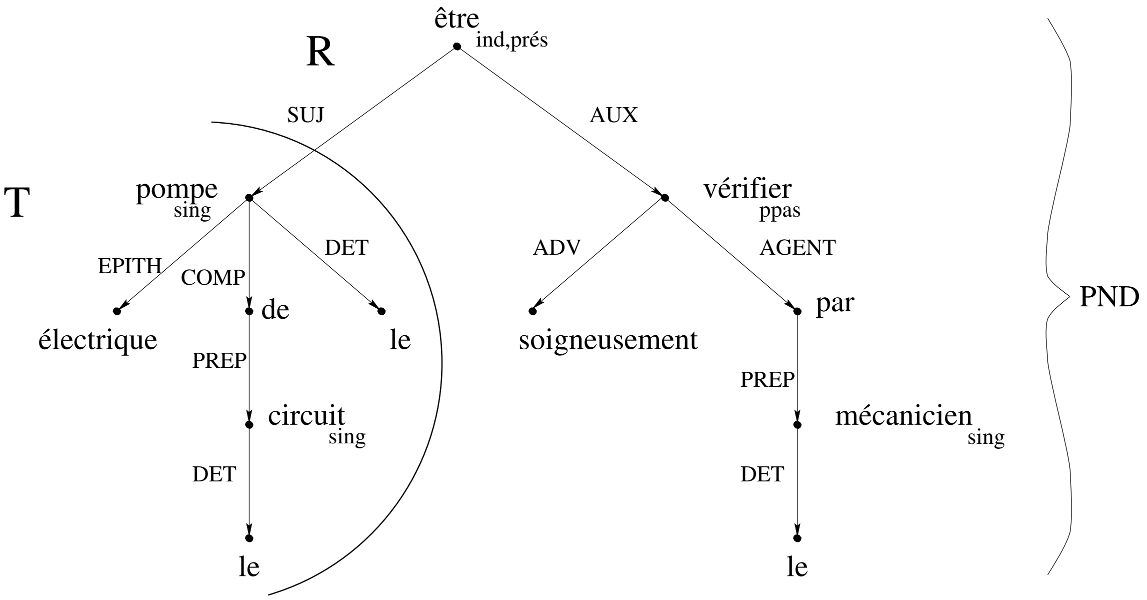


FIG. 2.5: Une représentation syntaxique de surface correspondant à la phrase *La pompe électrique du circuit est vérifiée soigneusement par le mécanicien*

Les nœuds de la structure syntaxique de surface sont étiquetés par les mots de la phrase. Ainsi les lexèmes « vides » (certaines prépositions, auxiliaires de temps...) qui n'étaient pas représentés au niveau profond le seront au niveau de surface. Les fonctions lexicales qui apparaissaient dans l'arbre profond seront remplacées par leur valeurs, les phrasèmes qui étaient représentés par un seul nœud seront décomposés en sous-arbres syntaxiques de surface et, finalement, les pronoms seront introduits.

Les branches de la structure syntaxique de surface sont étiquetées par des *relations syntaxiques de surface*. A la différence des relations profondes, les relations de surface ne sont pas universelles : toute langue propose une série de relations qui lui est propre. On trouvera dans [Mel'čuk & Pertsov, 1987] une tentative d'inventaire des relations syntaxiques de surface de l'anglais. Un travail analogue quoique non exhaustif pour les relations syntaxiques de surface du français est proposé dans [Béringer, 1988].

Les *RSyntS* des figures 2.6 et 2.7 correspondent à la *RSyntP* de la figure 2.4. Les différences de nature entre nœuds et branches des structures profondes et surfaciques induisent des différences structurelles entre arbres profonds et arbres de surface. L'auxiliaire de temps est maintenant représenté par un nœud qui constitue la racine de l'arbre syntaxique, la détermination nominale

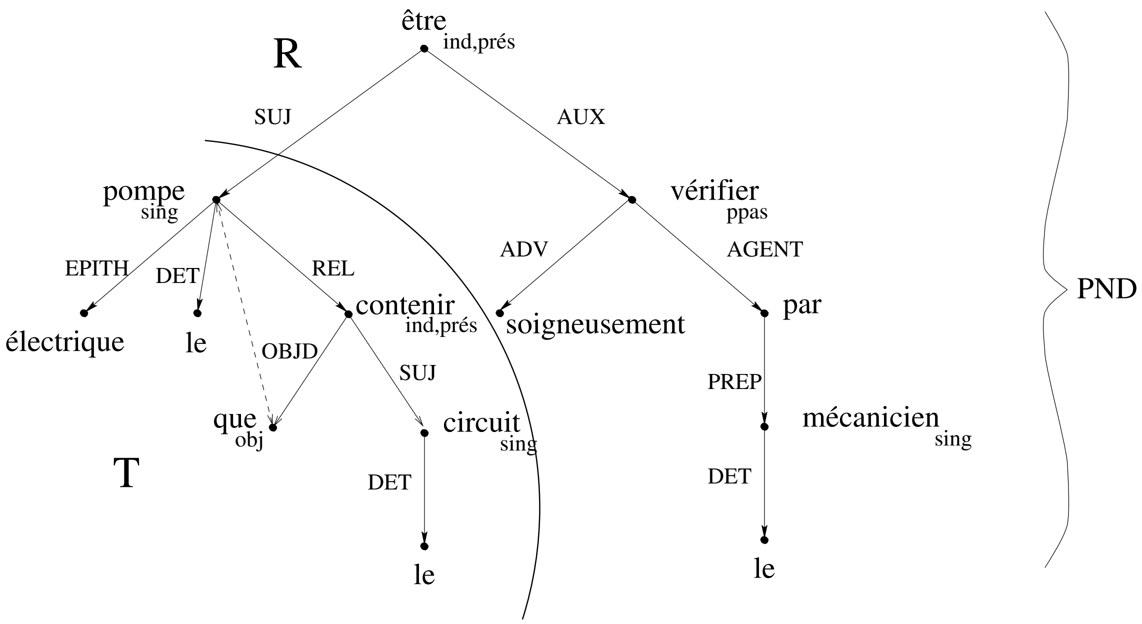


FIG. 2.6: Une représentation syntaxique de surface de la phrase *La pompe électrique que le circuit contient est vérifiée soigneusement par le mécanicien*

représentée au niveau précédent par des traits est maintenant réalisée par des articles. Les deux *RSyntS* se distinguent par la réalisation morpho-syntaxique du lexème profond *contenir* : subordonnée participiale dans un cas, et subordonnée relative dans l'autre.

La représentation morphologique profonde

Contrairement aux deux niveaux précédents qui proposaient des structures arborescentes, la représentation morphologique profonde se présente sous une forme linéaire. Elle est constituée de deux sous-structures, une structure morphologique profonde et une structure prosodique profonde. La structure morphologique profonde consiste en une séquence linéaire des lexèmes de la phrase enrichis de tous leurs traits morphologiques. L'ordre linéaire et les marqueurs morphologiques sont les deux seuls moyens de représentation disponibles à ce niveau. La structure arborescente de la *RSyntS* ainsi que les structures anaphoriques et communicatives seront réinterprétées en terme d'agencement linéaire des mots et de marqueurs morphologiques. La structure prosodique, que l'on ne représentera pas ici, indique les pauses, intonations et autres marqueurs prosodiques.

Les trois *RSyntS* du paragraphe précédent donneront naissance à trois structures morphologiques profondes :

$Le_{sing,fem}$ $pompe_{sing,fem}$ $électrique_{sing,fem}$ de $le_{sing,masc}$ $circuit_{sing,masc}$ $être_{ind,pres,3,sing}$ $vérifier_{ppas,sing,fem}$ $soigneusement$ par $le_{sing,masc}$ $mécanicien_{sing,masc}$.

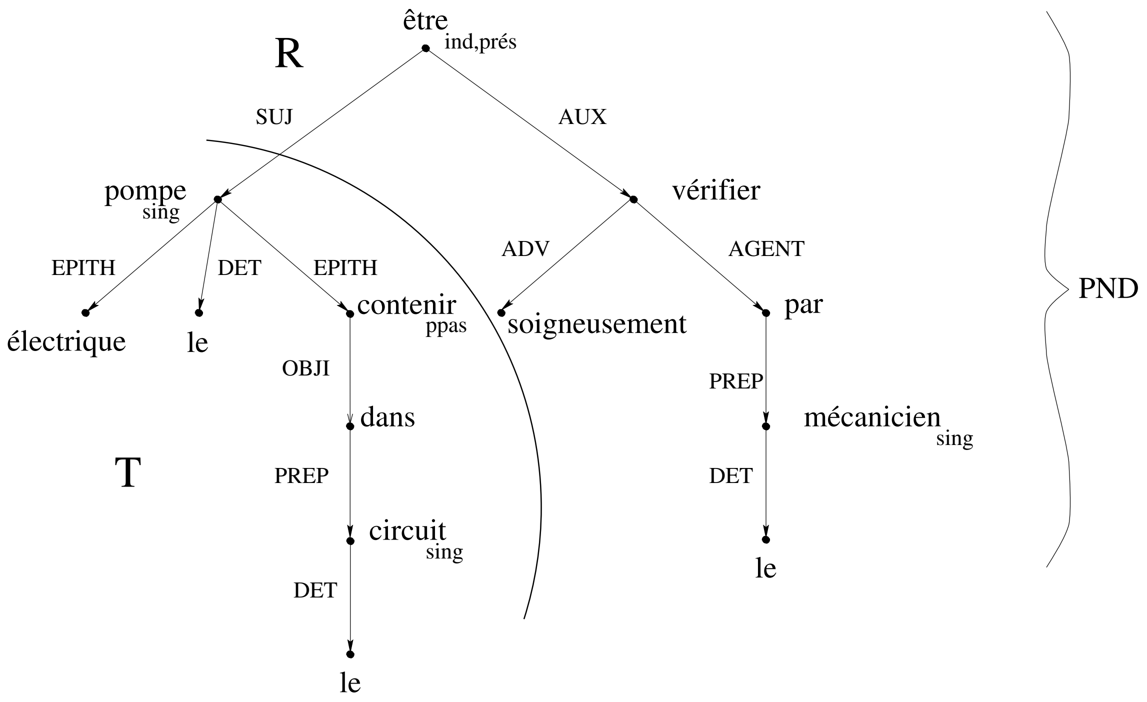


FIG. 2.7: Une représentation syntaxique de surface de la phrase *La pompe électrique contenue dans le circuit est vérifiée soigneusement par le mécanicien*

$Le_{sing,fem}$ $pompe_{sing,fem}$ $électrique_{sing,fem}$ $contenir_{ppas,sing,fem}$ $dans$ $le_{sing,masc}$ $circuit_{sing,masc}$ $être_{ind,pres,3,sing}$ $vérifier_{ppas,sing,fem}$ $soigneusement$ par $le_{sing,masc}$ $mécanicien_{sing,masc}$.

$Le_{sing,fem}$ $pompe_{sing,fem}$ $électrique_{sing,fem}$ que $le_{sing,masc}$ $circuit_{sing,masc}$ $contient_{ind,pres,3,sing}$ $être_{ind,pres,3,sing}$ $vérifier_{ppas,sing,fem}$ $soigneusement$ par $le_{sing,masc}$ $mécanicien_{sing,masc}$.

Bien que l'ordre linéaire des trois exemples précédents soit presqu'entièrement contraint par la structure syntaxique et la structure communicative, certains choix linéaires sont encore possibles ; la première *RMorphP* aurait pu prendre une des deux formes suivantes différant par la position de l'adverbe *soigneusement* :

$Le_{sing,fem}$ $pompe_{sing,fem}$ $électrique_{sing,fem}$ de $le_{sing,masc}$ $circuit_{sing,masc}$ $être_{ind,pres,1,sing}$ $soigneusement$ $vérifier_{ppas,sing,fem}$ par $le_{sing,masc}$ $mécanicien_{sing,masc}$.

$Le_{sing,fem}$ $pompe_{sing,fem}$ $électrique_{sing,fem}$ de $le_{sing,masc}$ $circuit_{sing,masc}$ $être_{ind,pres,1,sing}$ $vérifier_{ppas,sing,fem}$ par $le_{sing,masc}$ $mécanicien_{sing,masc}$
$soigneusement$.

De plus, la troisième *RMorphP* aurait pu être le siège d'une postposition du sujet de la relative :

$Le_{sing,fem}$ $pompe_{sing,fem}$ $électrique_{sing,fem}$ que $contient_{ind,pres,3,sing}$ $le_{sing,masc}$ $circuit_{sing,masc}$ $être_{ind,pres,3,sing}$ $vérifier_{ppas,sing,fem}$ $soigneusement$ par $le_{sing,masc}$ $mécanicien_{sing,masc}$.

La représentation morphologique de surface

C'est le niveau de représentation le plus proche du texte écrit. Il reprend les structures du niveau précédent mais remplace, au sein de la *structure morphologique de surface*, les représentations morphologiques profondes des mots par les mots eux-mêmes, donnant les trois phrases suivantes :

La pompe électrique du circuit est vérifiée soigneusement par le mécanicien.

La pompe éléctrique contenue dans le circuit est vérifiée soigneusement par le mécanicien.

La pompe éléctrique que le circuit contient est vérifiée soigneusement par le mécanicien.

Bien que les trois phrases précédentes soient grammaticales, elles ne sont pas équivalentes d'un point de vue stylistique : la première semble exprimer son contenu sémantique de la façon la plus simple, telle qu'elle apparaîtrait probablement dans un manuel. Un tel contrôle stylistique n'est pas prévu dans le cadre de la *TST*.

2.1.2 Le dictionnaire explicatif combinatoire

Nous n'avons pas explicité le détail des transitions entre les niveaux présentés décrits ci-dessus, il n'est pas difficile cependant de se convaincre du rôle essentiel que joue le lexique dans ces dernières. C'est en effet lui qui permet de remplacer des sous-réseaux sémantiques par des lexème, lors de la lexicalisation et c'est naturellement à son niveau que sont représentées les valeurs des fonctions lexicales. Ces connaissances seront représentées au sein d'un lexique d'une structure particulière appelé le *Dictionnaire Explicatif Combinatoire* (*DEC*) [Mel'čuk et al., 1984, Mel'čuk et al., 1987, Mel'čuk et al., 1992].

Le *DEC* est décomposé en articles, chaque article étant consacré à un lexème ou à un phrasème. Un lexème sera représenté par un mot suivi d'un chiffre indiquant une acception particulière de ce dernier. Un article comporte trois zones :

- zone sémantique

  Elle établit une correspondance entre un lexème et un réseau sémantique représentant *grosso modo* la définition du lexème[7]. Les nœuds du réseau sémantique sont étiquetés soit par des lexèmes soit par des variables. Nous avons représenté, dans la figure 2.8, le réseau associé au lexème *aider2*, extrait de [Mel'čuk & Polguère, 1987].

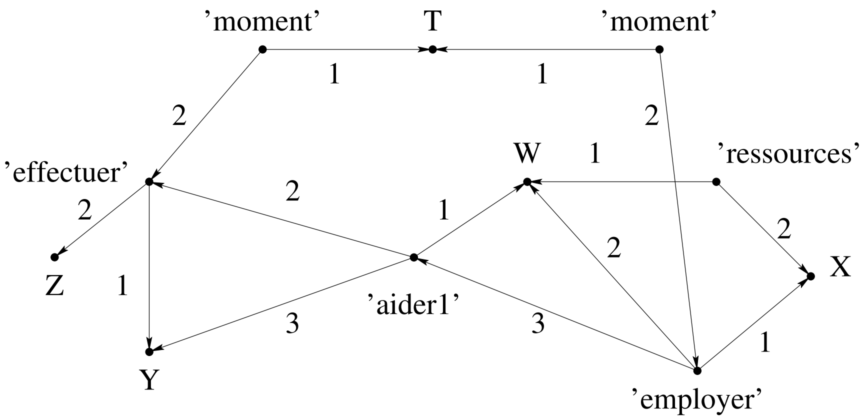


FIG. 2.8: Le réseau sémantique associé au lexème *aider2*

Le réseau sémantique de la figure 2.8 peut être lu de la façon suivante : ≪ Y étant en train d'effectuer une action Z, X emploie ses ressources W dans le but de causer que W aident1 Y à effectuer Z ; l'utilisation par X de ses ressources et la réalisation de Z par Y sont simultanés≫. Une occurrence du lexème *aider2* apparaît dans la phrase *Alcide a aidé Mordecaï à passer son*

[7]Il faut noter que dans les exemplaires existants du *DEC*, la définition des lexèmes n'est pas donnée sous la forme d'un réseau sémantique mais sous une forme propositionnelle proche de la lecture du réseau de la figure 2.8 présentée ci-dessus. On trouvera dans [Polguère, 1992] quelques remarques sur la correspondance réseau sémantique ⟷forme propositionnelle.

bac par ses conseils avertis et une occurrence d'*aider1* dans *La lumière aide les plantes dans leur croissance*. Les réseaux sémantiques apparaissant dans la zone sémantique répondent à deux critères :

- Le critère de *décomposition*, imposant qu'un lexème soit défini en termes de lexèmes sémantiquement plus « *simples* », afin d'éviter des phénomènes de circularité. On ne peut définir le lexème *aider1* en fonction d'*aider2* et, inversement, *aider2* en fonction d'*aider1*.
- Le critère du *bloc maximal* qui impose que dans un réseau sémantique, tout sous-réseau correspondant à un lexème soit remplacé par ce dernier. Ce principe de décomposition permet de faire apparaître clairement les relations sémantiques entre lexèmes tout en facilitant la lecture des réseaux sémantiques. C'est en vertu de ce principe que *aider1* n'est pas décomposé en un réseau de sémantèmes plus simples dans la définition de *aider2*.

• zone de combinatoire syntaxique

Elle permet d'une part de faire le lien entre les actants sémantiques et les actants syntaxiques profonds d'un lexème, et d'autre part d'indiquer les moyens de surface utilisés pour réaliser les actants syntaxiques profonds. Elle se présente sous la forme d'un tableau, appelé *schéma de régime* du lexème. On trouvera dans la première ligne du schéma de régime, les variables apparaissant dans le réseau sémantique ou dans la forme propositionnelle de la zone sémantique. La seconde ligne énumère les actants syntaxiques du lexème. La correspondance entre actants syntaxiques et actants sémantiques se trouve ainsi réalisée : dans une même colonne seront représentés un actant sémantique et l'actant syntaxique qui lui correspond. Apparaîtront ensuite dans chaque colonne les moyens syntaxiques de surface permettant de réaliser l'actant syntaxique (partie du discours, préposition et autres contraintes morphologiques). On trouvera, figure 2.9, le schéma de régime du lexème *aider2* emprunté à [Mel'čuk & Polguère, 1987].

X	Y	Z	W
I	II	III	IV
1. N	1. N	1. à N 2. à V 3. dans N 4. pour N 5. pour V	1. de N 2. avec N 3. par N 4. Adv

FIG. 2.9: Schéma de régime du lexème *aider2*

Dans la phrase d'exemple *Alcide aide Mordecaï à passer son bac par ses conseils avertis*, le sujet de la phrase, *Alcide*, correspond à l'actant profond numéro I et au nœud étiqueté X dans le graphe sémantique de la figure 2.8. Le substantif *conseil* correspond à l'actant profond IV et au nœud étiqueté W dans le graphe sémantique. Sa réalisation de surface correspond à la troisième ligne de la colonne IV : le syntagme prépositionnel *avec N*.

Peuvent être associées à un schéma de régime des restrictions sur la cooccurrence et la réalisation des actants : conditions sur la nature des actants, impossibilité pour un actant particulier d'apparaître avec ou sans un autre...

- zone de combinatoire lexicale

  Elle regroupe l'ensemble des fonctions lexicales définies pour le lexème. On trouvera, entre autres, dans l'entrée de *aider2* les fonctions lexicales permettant de lier *aider2* à des synonymes, un converse et des antonymes.

  $Syn(aider2)$: secourir, seconder, assister2

  $Conv_{213}(aider2)$: profiter

  $Anti(aider2)$: nuire, entraver, empêcher

2.1.3 Variété des faits de paraphrase

La description des différents niveaux de représentation des énoncés dans la *TST* permet d'expliquer le mécanisme de création de paraphrase. Au fur et à mesure que l'on se rapproche de la *RMorphS*, les réalisations d'une *RSém* se multiplient.

La situation est résumée dans la figure 2.10, qui laisse apparaître deux moyens de création de représentations paraphrastiques : les moyens *dérivationnels*, lorsque plusieurs structures sont dérivées d'une seule, à la faveur d'une transition entre niveaux de représentation, et des moyens *transformationnels*, lorsque plusieurs structures sont dérivées d'une seule sans changement de niveau de représentation. On trouvera dans la première catégorie les trois transitions *RSém* $\longrightarrow$ *RSyntP*, *RSyntP* $\longrightarrow$ *RSyntS* et *RSyntS* $\longrightarrow$ *RMorphP*. Le seul représentant de la seconde catégorie est la transformation *RSyntP* $\longrightarrow$ *RSyntP*, réalisée par le *système de paraphrase* de la *TST*. Ce dernier constitue le cœur de notre système de reformulation, il sera décrit en 2.1.4

Cette prolifération de structures s'explique par le fait qu'à chaque transition, un certain nombre de *choix* peuvent être effectués pour la construction de la structure suivante, multipliant ainsi le nombre de structures. Nous allons passer en revue les différents choix qui peuvent être faits lors de chacune de ces quatre phases.

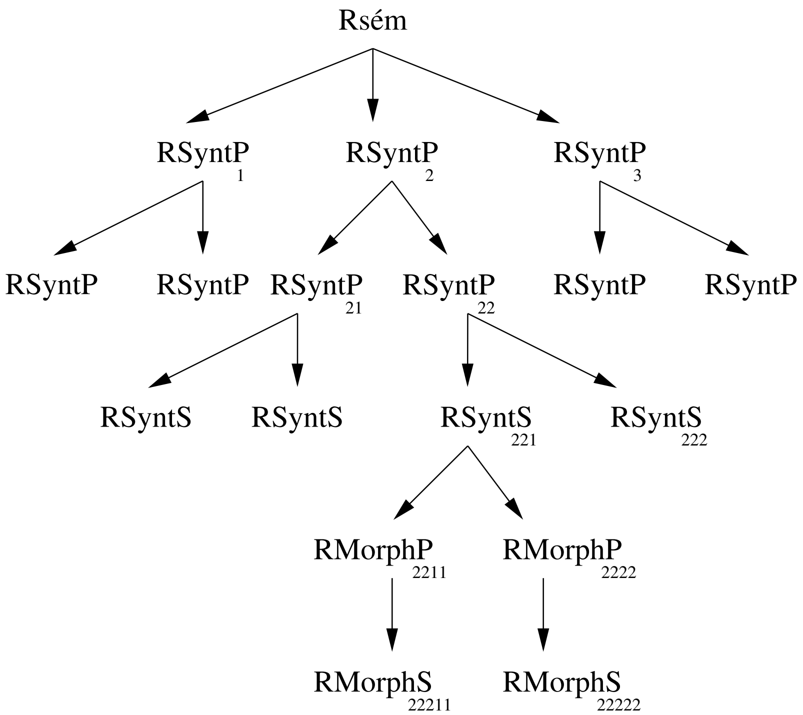


FIG. 2.10: Différentes réalisations d'une *RSém*

- La transition *RSém* $\longrightarrow$*RSyntP*

  C'est à ce niveau que la majorité des choix syntaxiques et lexicaux fondamentaux sont effectués, par l'intermédiaire de l'opération de lexicalisation. Du point de vue lexical, le réseau sémantique composé de sémantèmes liés entre eux est « découpé » en sous-réseaux correspondant à des lexèmes plus ou moins complexes. Comme nous avons pu le constater dans les exemples précédents, lorsque différents découpages sont possibles, plusieurs *RSyntP* sont créées. Les choix lexicaux vont influer sur la structure syntaxique, dans la mesure où le choix d'un lexème particulier va imposer certaines contraintes d'ordre syntaxique. Le choix d'un verbe, par exemple, va imposer une structure actancielle particulière. Il faut noter ici l'importance du degré de décomposition de la *RSém.* Plus cette dernière sera fine (composée de sémantèmes simples), plus les possibilités de lexicalisation seront nombreuses.

- La transformation *RSyntP* $\longrightarrow$*RSyntP*

  La transition précédente a donné naissance à une ou plusieurs *RSyntP* particulières appelées *RSyntP canoniques.* C'est sur ces dernières que va s'appliquer le système de paraphrase qui va permettre de dériver d'autres *RSyntP* synonymes. Cette transformation de *RSyntP* ne remet pas profondément en cause les choix lexicaux effectués lors de la lexicalisation, mais autorise certaines transformations lexicales locales. C'est ainsi qu'un lexème pourra être remplacé par un synonyme, un antonyme ou un converse. Certaines transformations perturberont un peu plus le paysage lexico-syntaxique, telles que le remplacement d'un verbe par une construction à verbe support.

- La transition *RSyntP* ⟶*RSyntS*

  Le choix principal effectué à ce niveau concerne la réalisation des fonctions lexicales : une fonction lexicale peut être réalisée de plusieurs façons, chaque choix pouvant donner naissance à une nouvelle *RSyntS*. Un second choix de moindre importance concerne les lexèmes vides, non représentés dans la *RSyntP* (Jacques aide son ami *à (pour)* obtenir ce poste).

- La transition *RSyntS* ⟶*RMorphP*

  L'unique choix paraphrastique qui est effectué à ce niveau concerne l'organisation linéaire de la phrase. Celle-ci est fortement déterminée par la syntaxe et la structure communicative de la phrase. Une certaine liberté d'ordre stylistique subsiste néamoins. Ces choix peuvent concerner, par exemple, l'ordre des actants et circonstants d'un verbe (*vendre X à Y pour Z francs vs vendre à Y X pour Z francs* ...).

2.1.4 Le système de paraphrase

Des transformations et transitions décrites ci-dessus, la seule dédiée exclusivement à la paraphrase est la transformation *RSyntP* ⟶*RSyntP* effectuée par le système de paraphrase. Ce dernier permet de générer, à partir d'une *RSyntP* plusieurs autres *RSyntP* synonymes. Il repose sur quatre composants distincts :

1. Les fonctions lexicales

   Les fonctions lexicales ont été définies ci-dessus comme moyen de formaliser des relations sémantiques entre lexèmes. Deux grandes classes de fonctions lexicales sont distinguées : les fonctions lexicales *paradigmatiques* et les fonctions lexicales *syntagmatiques*.

   Les fonctions lexicales paradigmatiques servent à associer un mot-clé à un ensemble de lexèmes qui partagent *dans le lexique* une composante sémantique non triviale avec le mot clé. La valeur de la fonction lexicale et le mot-clé ne forment pas habituellement un syntagme. Par exemple la fonction *Syn* mettant en correspondance deux synonymes est une fonction lexicale paradigmatique, de même que les fonctions liant deux verbes converses tels que *donner* et *recevoir*.

   Les fonctions lexicales syntagmatiques servent à formaliser une relation sémantique (éventuellement nulle) entre deux lexèmes L1 et L2 qui est réalisée *dans la chaîne textuelle* d'une façon non prévisible. Entrent par exemple dans cette catégorie les fonctions lexicales liant un nom à ses verbes supports.

   La *TST* recense une soixantaine de fonctions lexicales ([Mel'čuk, 1988b]).

2. Les règles lexicales de paraphrase

   Elles représentent soit des (quasi) équivalences sémantiques, soit des implications sémantiques formulées en termes de fonctions lexicales. Plus concrètement, elles établissent que tels noeuds lexicaux de la *SSyntP* (un ou deux) peuvent être remplacés *salva significatione* par tels autres noeuds (également un ou deux), les derniers étant les fonctions lexicales des premiers ou bien les premiers et les derniers étant les fonctions lexicales du même mot-clé. Une règle lexicale simple établissant une correspondance entre deux verbes liés par une fonction lexicale conversive, tels que *acheter* et *vendre* se présentera de la façon suivante :

$$\begin{array}{ccc} X & & Y \\ V & \Longleftrightarrow & Conv_{132}(V) \end{array}$$

   Une règle plus complexe dite de fission à verbe support reliera un verbe et une construction à verbe support, tels que *mesurer* et *effectuer une mesure* :

$$\begin{array}{ccccc} X & & Y & & Z \\ V & \Longleftrightarrow & S_0(V) & \xleftarrow{II} & Oper_1(S_0(V)) \end{array}$$

   La fonction lexicale S_0 apparaissant dans la règle est une fonction lexicale paradigmatique reliant un verbe et une nominalisation de ce dernier et $Oper_1$ une fonction lexicale syntagmatique liant un nom à certains de ses verbes opérateurs. La règle appliquée au couple *mesurer*, *effectuer une mesure* se présentera de la façon suivante :

$$\begin{array}{ccccc} X & & Y & & Z \\ mesurer & \Longleftrightarrow & mesure & \xleftarrow{II} & effectuer \end{array}$$

3. Les règles syntaxiques de paraphrase

   Elles effectuent les restructurations requises par les changements lexicaux. Chaque règle lexicale est associée à une séquence de règles syntaxiques. Les règles syntaxiques se présentent sous la forme de transformations d'arbres de dépendances étiquetés. Ces transformations sont introduites et analysées dans le cadre des *dendrogrammaires* [Gladkij & Mel'čuk, 1975] sur lesquelles nous reviendrons en 3.5. Nous nous contenterons ici de donner, dans la figure 2.11, un exemple de règle dite de réétiquetage des branches actantielles, nécessaire lors du remplacement d'un verbe par un converse pour intervertir les actants des deux structures et dans la figure 2.12, un exemple de règle dite de fission, utilisée lors de remplacement d'un verbe par une construction à verbe support.

   Les conventions d'écriture sont simples : les variables X, Y et Z identifient des noeuds affectés par des substitutions lexicales (dans les règles lexicales), permettant de faire le lien entre règles syntaxiques et règles lexicales.

La décomposition des règles syntaxiques et des règles lexicales dans la *TST* est principalement guidée par un souci d'économie, les mêmes règles syntaxiques pouvant être utilisées

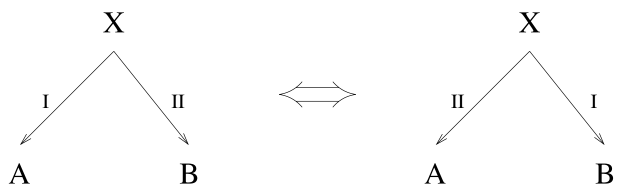


FIG. 2.11: Une règle de réétiquetage des branches actantielles

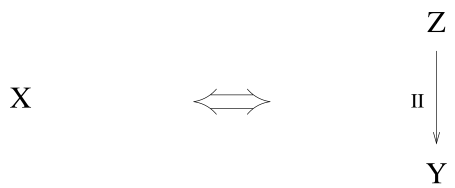


FIG. 2.12: Une règle de fission

dans plusieurs règles lexicales.

4. Les méta-règles

   L'application des règles syntaxiques de paraphrase est soumise à des conditions générales représentées sous la forme de méta-règles. Deux méta-règles sont proposées dans [Mel'čuk, 1988b] :

Méta-règle1 Lors de l'application d'une règle syntaxique de paraphrase R à une *SSyntP*, tout nœud de cette structure qui n'est pas mentionné dans R reste attaché à son gouverneur par la même dépendance sytaxique profonde, sauf mention expresse du contraire dans une autre méta-règle.

Méta-règle2 Si, lors de l'application d'une règle syntaxique de paraphrase R à une *SSyntP*, un nœud X est remplacé par une structure $Z \rightarrow Y$, alors les dépendants de X qui ne sont pas mentionnés dans R sont distribués entre Y et Z de la façon suivante :

- L'actant I de X reste auprès du sommet du nouveau sous-arbre, c'est-à-dire auprès de Z.
- Les actants II à VI passent de X à Y.
- Les modificateurs circonstanciels restent auprès de Z.
- Les modificateurs qualitatifs de X passent à Y, si Z est un verbe support ou bien ils peuvent soit rester auprès de Z, soit passer à Y si Z est un verbe sémantiquement plein.

Une règle de paraphrase se présente comme une règle lexicale accompagnée d'une séquences de règles syntaxiques. La règle de paraphrase permettant de remplacer un verbe par une structure à verbe support sera composée de la règle lexicale vue ci-dessus accompagnée de la règle syntaxique de fission de la figure 2.12[8].

Comme nous l'avons déja remarqué, les règles de paraphrases réalisent des modifications locales telles que le remplacement d'un mot ou d'une structure simple (deux mots liés par une dépendance syntaxique). Le système de paraphrase est composé d'un ensemble de ces règles locales. La dérivation de nouvelles *RSyntP* paraphrastiques est réalisée par application d'une ou plusieurs règles à une *RSyntP* existante.

2.2 Reformulation

Le processus de génération de paraphrases que nous avons présenté dans la section précédente ne concernait que la moitié du processus de paraphrasage d'une phrase : la transition Sens

[8] On pourra remarquer que la distinction entre règles syntaxiques et règles lexicales de paraphrase n'est pas toujours bien respectée dans la *TST* : la règle de fission à verbe support présentée ci-dessus n'est pas purement lexicale dans la mesure où elle introduit en plus de deux fonctions lexicales une dépendance entre ces deux dernières, empiétant ainsi sur le domaine des règles syntaxiques.

⟶Texte. Le processus complet suppose une transition préalable : la transition Texte ⟶Sens. Le déroulement d'un processus de paraphrasage s'organise donc naturellement en deux phases : une phase d'analyse (Texte ⟶Sens), à l'issue de laquelle une ou plusieurs *RSém* sont construites, et une phase de génération (Sens ⟶Texte) à l'issue de laquelle plusieurs phrases synonymes sont générées. La reformulation nécessite par conséquent de remonter jusqu'au sens d'une phrase pour la reformuler.

A la différence de la *TST*, nous ne cherchons pas à générer toutes les paraphrases d'une phrase donnée, mais uniquement certaines, compatibles avec les règles d'une langue contrôlée. La reformulation peut alors être vue comme une *génération contrôlée* de paraphrases. Le contrôle de la paraphrase va consister à effectuer, à chacune des étapes de la transition Sens ⟶ Texte qui ont été identifiées ci-dessus, des choix compatibles avec les prescriptions d'une langue contrôlée.

Les deux sections suivantes traitent des deux points soulevés ci-dessus. La première s'intéressera à la « profondeur » des reformulations et aboutira à une remise en cause du schéma Texte ⟶Sens ⟶Texte. La seconde traitera du contrôle de la paraphrase et du problème de la *détection* de constructions interdites par une langue contrôlée.

2.2.1 Profondeur de la reformulation

La génération de paraphrases en partant d'une *RSém*, n'est pas la seule possible : il est envisageable de générer des paraphrase d'une phrase P à partir d'une représentation moins profonde de cette dernière. Si notre interêt se limitait, par exemple, à des phénomènes de variation d'ordre linéaire dans la phrase, une paraphrase à partir du niveau syntaxique de surface pourrait suffire car c'est lors de la transition *RSyntS* ⟶*RMorphP* que les choix d'ordre linéaire sont effectués. Dans les deux cas précédents, la génération de paraphrases est effectuée à partir d'une représentation commune, un *invariant paraphrastique*, qui est dans un cas la *RSém* et dans le second, la *RSyntS*. Une telle architecture rappelle des modèles de traduction automatique où une phrase à traduire est d'abord représentée à un niveau indépendant de la langue (interlingua), pour ensuite être regénérée dans une langue cible. Dans le cas de la traduction, une même phrase possède une représentation unique dans plusieurs langues différentes et dans le cas de la reformulation, plusieurs énoncés paraphrastiques partagent une représentation commune : la *RSém*. Ce type de modèles s'oppose généralement à un second, dit « à transfert », où la phrase à traduire est représentée à un certain niveau d'abstraction non indépendant de la langue. A ce niveau est effectué le transfert, qui consiste à construire une représentation de même niveau d'abstraction dans la langue cible. Cette dernière permettra de générer la phrase dans la langue cible. [Hutchins & Somers, 1992] avancent deux avantages majeurs des systèmes « à transfert » sur les systèmes basés sur un invariant : la difficulté de concevoir un niveau de représentation indépendant de la langue, et la complexité des phases d'analyse et de génération entre la phrase et ce niveau de représentation abstrait.

A l'instar de la traduction automatique, une alternative « à transfert » existe pour la reformulation dans le cadre de la *TST*. Elle consiste à effectuer un transfert au niveau syntaxique profond. Le transfert lui-même est effectué par le système de paraphrase qui, comme on l'a vu, permet de dériver plusieurs *RSyntP* à partir d'une seule. Nous avons représenté, figure 2.13, les deux architectures, à gauche l'architecture basée sur un invariant paraphrastique (*RMorphS* $\longrightarrow$ *RSém* $\longrightarrow$ *RMorphS'*) et à droite l'architecture à transfert (*RMorphS* $\longrightarrow$ *RSyntP* $\longrightarrow$ *RSyntP'* $\longrightarrow$ *RMorphS'*).

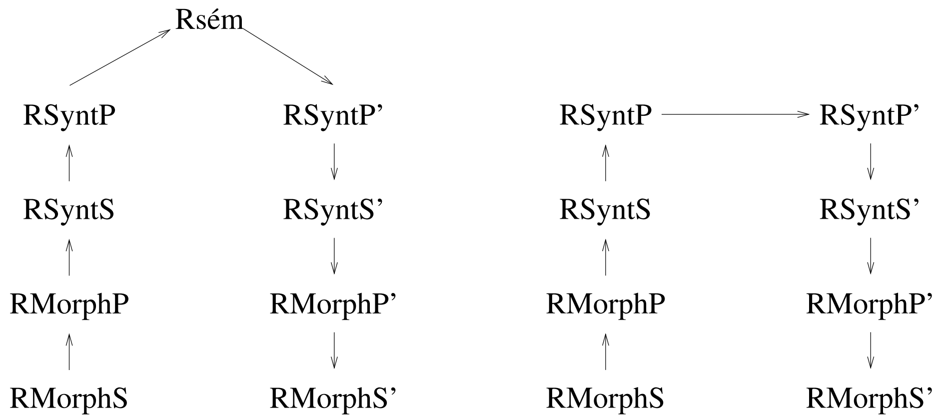


FIG. 2.13: Deux modèles de reformulation

C'est l'architecture à transfert que nous avons choisi d'adopter. Une telle approche ne nous permet pas de bénéficier de toute la puissance paraphrastique de la *TST* car c'est, comme nous l'avons vu, au niveau de la transition *RSém* $\longrightarrow$*RSyntP* que se situe une bonne partie de la capacité paraphrastique du modèle. Le choix de l'architecture à transfert est motivé tant par des considérations propres à notre application que par des considérations propres à la *TST*.

Du point de vue de notre application, les choix paraphrastiques effectués lors de la transition *RSém* $\longrightarrow$*RSyntP* concernent principalement le découpage d'un réseau sémantique en lexèmes plus ou moins complexes, effectué lors de l'opération de lexicalisation. Or, dans le type de textes que nous traitons et pour le type de reformulations auxquelles nous nous intéressons, une telle redistribution sémantico-lexicale n'est pas toujours souhaitable. En effet, les phrases à reformuler ont été élaborées par un rédacteur humain, qui a procédé à une première distribution sémantico-lexicale, conforme aux habitudes rédactionnelles d'un domaine technique. Les actions utilisées dans les textes correspondent à des classes d'actions particulières, les entités représentées correspondent généralement à des équipements techniques répertoriés. Une perturbation de ce découpage déjà effectué par le rédacteur a peu de chance d'aboutir à un redécoupage qui soit d'une part en adéquation avec un domaine technique et d'autre part meilleur que celui proposé par le rédacteur. Ce n'est pas à ce niveau que nous envisageons la reformulation, mais à un niveau moins profond, concernant plutôt des aspects terminologiques et des choix syntaxiques, reformulation qu'il semble plus adéquat d'effectuer au niveau syntaxique profond.

Il reste que les textes ne sont pas composés exclusivement de termes techniques, et, sans modifier ces derniers, la possibilité d'agir sur le découpage sémantico-lexical des aspects « non-techniques» des textes présente un interêt indiscutable. On pourra reprendre en guise d'illustration l'exemple de la section précédente, dans lequel deux réalisations textuelles d'un sous-réseau sémantique étaient possibles (*la pompe électrique du circuit vs la pompe électrique contenue dans le circuit*). Lorsque l'on se restreint à une paraphrase de niveau syntaxique profond, la génération de ces deux formes alternatives n'est pas possible. Une solution consiste à ne pas décomposer, au niveau sémantique, les termes techniques. Ces derniers seraient alors représentés par des sémantèmes complexes mais non décomposables, en correspondance bi-univoque avec des lexèmes du vocabulaire technique. Les autres parties « non-techniques » des textes seraient, elles, décomposées en sémantèmes plus simples pouvant donner lieu à des lexicalisations diverses. Nous aurions ainsi des *RSém* à différents niveaux de granularité, pouvant donner naissance à de multiples *RSyntP* préservant l'intégrité du vocabulaire technique. Une telle solution reste néanmoins confrontée aux problèmes abordés ci-dessous.

Du point de vue de la *TST*, la transition *RSém* $\longrightarrow$*RSyntP* suppose d'une part une décomposition des lexèmes en réseaux sémantiques et, d'autre part, des règles de transition permettant de transformer un réseau sémantique en un arbre lexico-syntaxique. Or ce sont des aspects encore mal maîtrisés dans la *TST* . La représentation d'un énoncé au niveau sémantique exige un langage précis et complet de représentation des réseaux sémantiques qui n'existe pas encore. Certains aspects, tels que la représentation des quantificateurs (analysée dans [Polguère, 1992]) sont difficiles ou impossibles à représenter. D'autre part, la transition *RSém* $\longrightarrow$*RSyntP* suppose, outre une définition de tous les lexèmes sous forme de réseaux sémantiques, des règles de correspondance entre sous-réseaux sémantiques et traits morphologiques, tel que le temps grammatical, qui sont encore mal décrits dans la *TST*.

Il est intéressant de remarquer que ces deux points correspondent, dans la problématique de la traduction automatique basée sur un invariant, aux problèmes mentionnés par [Hutchins & Somers, 1992], à savoir, la définition du langage permettant de représenter l'invariant et les problèmes de transition entre l'invariant et une représentation lexico-syntaxique.

2.2.2 Détection et correction

Le choix d'une architecture de type « transfert » que nous avons évoqué ci-dessus va induire une phase supplémentaire dans le processus de reformulation : la détection de constructions lexico-syntaxiques interdites. En effet, dans le schéma *RMorphS* $\longrightarrow$*RSém* $\longrightarrow$*RMorphS* ′, l'existence d'un invariant paraphrastique (la *RSém*) annule la nécessité de recourir à une phase de détection. Il n'est en effet pas nécessaire de détecter dans la *RSém* des structures interdites et de les remplacer par des structures autorisées, il « suffit », de construire à partir de la *RMorphS* une *RSém* et de générer, à partir de cette dernière, grâce à l'utilisation d'une *grammaire prescriptive*, une *RMorphS* exempte de constructions interdites. On ne se soucie pas dans ce cas de savoir si la phrase de départ comportait des constructions interdites : si c'était le cas, elles auront disparu dans la *RMorphS* générée grâce au recours à la grammaire prescriptive. L'invariant paraphrastique (la *RSém*) correspond à des phrases interdites ainsi qu'à des phrases autorisées.

Il en est autrement du modèle « à transfert », qui n'est plus basé sur un invariant paraphrastique. La transformation de la *RSyntP* exige de *détecter* dans cette dernière des configurations interdites pour les remplacer par des constructions autorisées, ceci grâce aux règles de paraphrase. Les deux phases de détection et de correction sont maintenant nécessaires au bon déroulement du processus de reformulation.

Dans le domaine de la reformulation, on retrouve avec [Hayashi, 1992] la distinction système à transfert *vs* système basé sur un invariant, sous l'opposition modèle de réécriture *vs* modèle de regénération. Le premier modèle est basé sur un ensemble de règles locales transformant une structure interdite en une structure autorisée. Le second prévoit l'existence d'un invariant et d'une grammaire prescriptive. Il s'agit là en fait de deux cas extrêmes car, dans le cas d'un modèle de réécriture, il est difficile de concevoir des règles applicables sans analyse préalable de la phrase à un certain niveau d'abstraction. Lorsque ce niveau d'abstraction est proche de la forme de surface, les capacités de reformulation du système seront restreintes. Lorsque le niveau d'abstraction est éloigné du niveau de surface, ce qui est souhaitable, la phase de regénération de la phrase devra être contrôlée à l'aide d'une grammaire prescriptive.

Le modèle de reformulation que nous proposons relève des deux approches : c'est un système de réécriture dans la mesure où il comporte des règles permettant de transformer des structures interdites (les règles de paraphrase), mais c'est aussi un système de regénération, du fait qu'il nécessite une grammaire prescriptive lors de la transition *RSyntP* ⟶*RMorphS*. En effet, après la phase de détection-correction, une *RSyntP* « corrigée » peut quand même donner naissance à une *RMorphS* non conforme à une langue contrôlée car les interdictions peuvent porter sur des aspects qui ne sont pas représentés au niveau syntaxique profond. Il est donc nécessaire de veiller à ce que la transition *RSyntP* ⟶*RMorphS* ne génère pas de constructions interdites. La phase de détection s'effectue au niveau syntaxique profond. La correction se décompose, elle, en deux étapes : une correction proprement dite au niveau de la transformation de la *RSyntP* et une regénération contrôlée à partir de la *RSyntP*. Un tel processus nécessite deux grammaires : une grammaire d'analyse générale (permettant d'analyser des phrases conformes ou non à une langue contrôlée) et une grammaire de génération prescriptive (ne générant que des phrases autorisées). La situation est résumée dans le schéma de la figure 2.14.

La mise en œuvre du processus de reformulation que nous allons détailler dans la suite de cette thèse présente une différence importante par rapport au schéma de la figure 2.14. Nous ne manipulerons en effet, à chaque niveau de représentation, que la structure principale associée à ce niveau. Nous ne manipulerons donc pas les structures rhétorique, communicatives et anaphoriques des différents niveau de représentation. Les représentations manipulées lors du processus de reformulation sont par conséquent incomplètes. Les raisons de cette restriction sont doubles. La première tient dans le manque de détail qu'offre la *TST* sur ces structures, qui ne sont souvent que citées. Nous avions remarqué en effet le flou qui entourait les définitions des structures rhétoriques, communicatives et prosodiques lors de la description des niveaux de représentations. La seconde

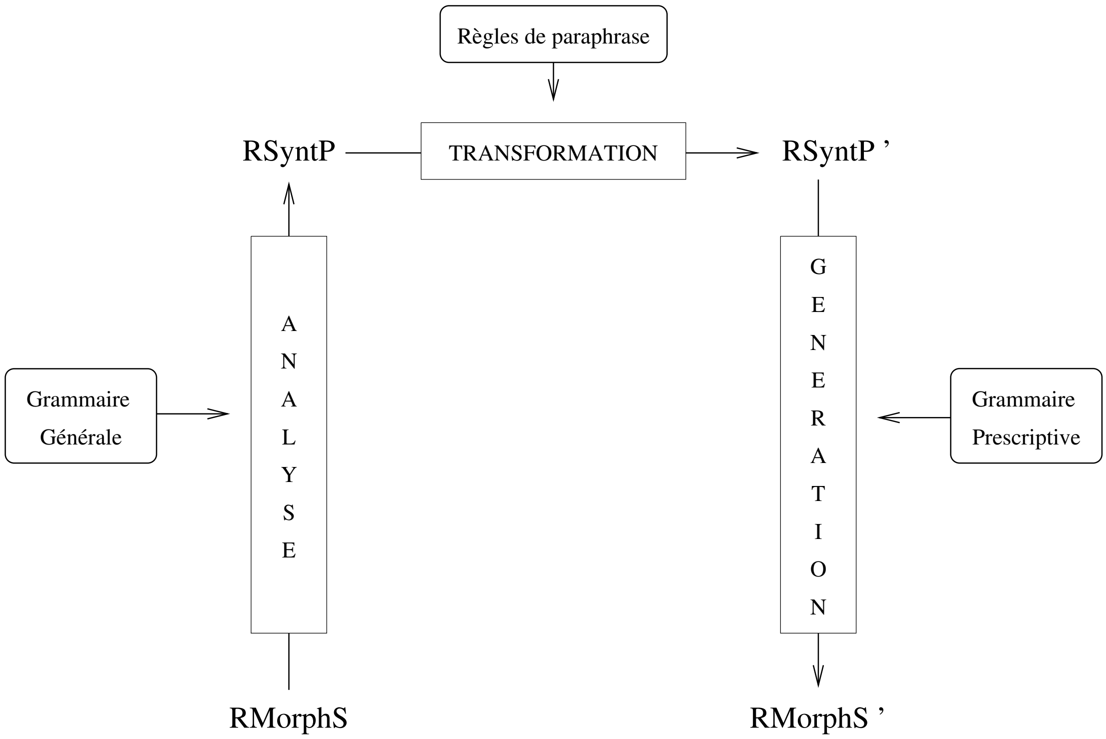


Fig. 2.14: Les trois étapes de la reformulation

raison concerne la difficulté, lors de la phase d'analyse, de construire les structures communicatives, prosodiques et anaphoriques de la phrase à reformuler. Nous avons donc décidé, pour ces deux raisons de nous limiter aux structures principales des différents niveaux de représentation. Cette restriction constitue une limite importante de notre processus de reformulation. En effet, le fait d'ignorer les dimensions communicatives, prosodiques et anaphoriques va accroître les risques de dérives sémantiques et provoquer, dans certains cas la génération de phrases agrammaticales. Nous verrons, en 6.3 comment limiter ces problèmes lors de la phase de regénération.

2.2.3 Deux problèmes théoriques

On peut dès à présent et indépendamment de toute considération de mise en œuvre du processus de reformulation, identifier deux problèmes théoriques auxquels nous allons être confronté.

Le premier est indépendant de la *TST* : il concerne la correspondance Texte $\longrightarrow$Sens$_i$. Comme le montre l'indice apparaissant au niveau du Sens, une phrase peut être ambiguë, auquel cas lui correspondent plusieurs *RSém*. Le problème consiste alors à savoir quelle est la bonne *RSém*, celle sur laquelle appliquer la reformulation. Le choix que nous avons fait de ne pas remonter jusqu'à la *RSém* d'une phrase ne change rien au problème, car à une phrase peuvent correspondre plusieurs *RSyntP*. La représentation et la prise en compte de connaissances extra-linguistiques peuvent certes réduire l'ambiguïté, on pensera notamment à un dictionnaire de groupes nominaux complexes propres à un domaine technique permettant de résoudre certains cas de rattachements prépositionnels ambigus. Le problème n'est pas pour autant résolu. Il est intéressant de remarquer que certains cas d'ambiguïtés n'empêchent pas la reformulation (*Jean a vu un homme avec un télescope* $\longrightarrow$*Jean a aperçu un homme avec un télescope*). Cela se comprend intuitivement par le fait que certaines reformulations n'exigent pas une « compréhension » profonde de la phrase à reformuler : une « compréhension » imparfaite est quelquefois suffisante. La prise en compte de tels cas suppose d'une part de recenser quelles ambiguïtés ne posent pas de problèmes pour quels types de reformulations et d'autre part, de savoir détecter automatiquement certains types d'ambiguïtés. Sans résoudre le problème de l'ambiguïté, une telle approche pourrait résoudre un certain nombre de cas. C'est malheureusement une piste que nous n'avons pas eu le temps d'explorer dans le cadre de ce travail.

Le second problème concerne l'exactitude sémantique des paraphrases produites. C'est principalement au niveau du système de paraphrase de la *TST* qu'il se situe. Les problèmes ont été regroupés dans [Mel'čuk, 1988b] sous deux rubriques :

- L'absence de précisions sémantiques

  Les équivalences établies par les règles lexicales de paraphrase (en termes de fonctions lexicales) le sont à un niveau abstrait, idéal. Linguistiquement, lorsque les fonctions lexicales sont remplacées par des termes de la langue, certains décalages peuvent apparaître. On peut voir là une manifestation de la nonchalance incontrôlée dont il était question en 2.1 : certaines nuances sémantiques non représentées provoquent des paraphrases incorrectes. Pour [Mel'čuk, 1988b], la paraphrase doit se faire sous le contrôle de la sémantique. Il faudrait vérifier, lors de l'application de règles de paraphrase, que les différences sémantiques entre les lexèmes introduits par les fonctions lexicales et les lexèmes remplacés ne provoquent pas d'erreurs ou de décalages de sens. Cette solution reste théorique : la *TST* ne fournit pas à l'heure actuelle les moyens d'effectuer ce type de vérification.

- L'absence de conditions d'application des règles de paraphrase

  Les règles de paraphrase de la *TST* permettent d'échanger, dans une *RSyntP*, deux expressions ≪ quasi ≫ identiques. Or l'applicabilité de ces règles est contrainte par plusieurs conditions syntaxiques. Ces conditions ne sont pas explicitées dans les règles de paraphrase, provoquant quelquefois la génération de paraphrases incorrectes. Le remplacement, par exemple, d'une construction à verbe support par un verbe sémantiquement plein peut poser dans certains contextes des problèmes, comme dans l'exemple suivant : *le mécanicien effectue une mesure précise de tension* ⟶ **le mécanicien mesure précise la tension.* La transformation de l'adjectif *précise* en adverbe *précisément* ou en groupe prépositionnel *avec précision* qui devrait accompagner la règle de paraphrase ainsi que le lien entre les deux règles n'est pas pris en compte dans la *TST*. Nous introduirons au chapitre 6 des mécanismes permettant d'effectuer, dans certains cas, ces modifications.

Chapitre 3

Un modèle de reformulation

Nous avons distingué dans le chapitre précédent les différents niveaux de représentation d'un énoncé que propose la *TST* et qui vont être mis en jeu lors des trois phases d'analyse, de transformation et de regénération du processus de reformulation. Nous ne nous sommes par contre pas penché sur les moyens grâce auxquels sont réalisées les transitions d'un niveau de représentation à l'autre, qui constituent l'objet de ce chapitre. La *TST* propose diverses règles, regroupées au sein de *composantes*, permettant d'effectuer les transitions entre niveaux de représentation. Nous analyserons en 3.1 les principes qui sous-tendent la définition de ces règles et nous verrons qu'ils ne sont pas toujours compatibles avec nos contraintes de mise en œuvre du processus de reformulation. Nous proposerons alors, en 3.2, une réorganisation des règles de transitions et de paraphrase de la *TST* sous la forme d'*arbres élémentaires*, de relations entre arbres élémentaires et d'opération de manipulation des arbres élémentaires. Ces transformations de l'organisation initiale des connaissances au sein de la *TST* vont notamment permettre une représentation simple et précise des règles d'écriture d'une langue contrôlée, décrite en 3.3. Le chapitre s'achèvera sur la description des différents modules permettant de réaliser, par manipulation d'arbres élémentaires, les trois phases d'analyse, de transformation et de regénération du processus de reformulation. Ce chapitre est introductif, son but est de poser les principes fondamentaux du modèle de reformulation, de ses éléments et des rapports qu'ils entretiennent mais laissera dans l'ombre de nombreux détails que les chapitres suivants dévoileront.

3.1 Les particularités du formalisme

La mise en œuvre du processus de reformulation requiert un modèle de représentation des connaissances linguistiques nécessaires à cette activité, en d'autres termes, un formalisme grammatical pensé et conçu pour la reformulation. Ce dernier se définit plus précisément à la lumière des trois étapes - analyse, transformation et regénération - que nous avons distinguées dans le processus de reformulation. Le formalisme que nous proposons doit représenter les connaissances permettant de :

1. Construire, lors de la phase d'analyse, toutes les *SSyntP* d'une phrase.

2. Modifier une *SSyntP* en accord avec certaines règles d'écriture d'une langue contrôlée durant la phase de transformation.

3. Générer à partir d'une *SSyntP* au moins une phrase compatible avec les règles de la langue contrôlée, lors de la regénération.

A chacun de ces buts correspond un module chargé de sa réalisation proprement dite : module d'analyse pour le premier, de transformation pour le second et de regénération pour le troisième. Le formalisme permet de représenter les connaissances linguistiques que les modules utiliseront pour réaliser les trois buts présentés ci-dessus. Nous n'associerons pas au formalisme une procédure permettant de générer, par composition, des phrases grammaticales : là n'est pas sa raison d'être.

La question se pose, au vu des trois buts formulés ci-dessus, de savoir si ces derniers sont vraiment différents des buts de la *TST*, ou plus exactement s'ils ne sont pas inclus dans les buts (plus généraux) de cette dernière. En effet, I. Mel'čuk écrit dans [Gladkij & Mel'čuk, 1975], et c'est le premier postulat de la *TST*, que la langue naturelle est considérée comme une application multi-multivoque entre un ensemble dénombrable de sens et un ensemble dénombrable de textes. Le second postulat propose que cette application soit représentée par un dispositif logique, c'est à dire par un modèle cybernétique ou fonctionnel de la langue. Détaillant ce postulat, I. Mel'čuk pose que par sa nature, un *Modèle Sens-Texte*[1] complet doit être dynamique : il assure le passage entre un sens donné et tous les textes qui expriment ce sens (ou bien entre un texte donné et tous les sens que ce texte exprime). Le troisième postulat, enfin, propose deux représentations intermédiaires entre sens et textes : un niveau syntaxique et un niveau morphologique. Les différences entre nos buts et ceux de la *TST*, énoncés ci-dessus, ne sont pas fondamentales. Le premier but que nous avons distingué (construction des *SSyntP* d'une phrase) s'intègre parfaitement dans le cadre de la transition Texte $\longrightarrow$Sens. Les deux derniers (transformation d'une *SSyntP* et regénération d'une phrase conforme aux règles d'une langue contrôlée) peuvent être rapprochés de la transition Sens $\longrightarrow$Texte. Ils s'en distinguent toutefois un peu en ajoutant à cette dernière des contraintes d'ordre stylistique, permettant de contrôler l'application des règles de paraphrase et de guider la

[1]Un Modèle Sens-Texte (ou *MST*) d'une langue L est l'application des principes de la Théorie Sens-Texte à L.

phase de génération d'une phrase à partir d'une *SSyntP*. Leur apport peut être vu comme l'ajout d'un module de « contrôle stylistique » à la transition Sens $\longrightarrow$Texte.

Des différences importantes existent néanmoins entre notre approche et celle de la *TST*. Elles sont partiellement données par I. Mel'čuk lui-même dans la suite du même texte, où l'attention est portée sur une distinction que l'on peut effectuer entre deux sous-modèles dans un *MST* :

1. Un système de règles purement linguistiques qui spécifient la correspondance entre sens et texte de façon statique.

2. Un système de règles procédurales dynamiques qui spécifient le procès de passage entre les sens et les textes, c'est à dire qui effectuent la correspondance $RSem_i \longleftrightarrow RPhonet_j$.

I. Mel'čuk souligne alors que : « *les règles procédurales manipulent des données présentées par des règles linguistiques ; ou, en d'autres termes, le deuxième système ne fonctionne qu'à la base du premier. Or nous croyons que le deuxième système de règles n'est pas spécifique à la linguistique : le même type général de règles procédurales est nécessaire partout où des règles emmagasinées (représentant des connaissances statiques) doivent être appliquées à la solution d'une tâche. Pour cette raison, nous faisons abstraction de ce deuxième système. Par conséquent, nous présentons le MST comme un système entièrement statique de correspondance entre sens élémentaires et textes élémentaires - si bien que, par exemple, le problème de l'ordonnancement des règles de correspondance ne se pose même pas.* »

Cette mise au point de I. Mel'čuk va nous permettre de mieux nous situer par rapport à la *TST*. Nous retrouvons la distinction que I. Mel'čuk fait entre règles linguistiques et règles procédurales dans la dissociation que nous avons opérée entre la représentation des connaissances nécessaires à la reformulation et les modules permettant de réaliser la reformulation. Mais, contrairement à I. Mel'čuk, nous portons un interêt particulier à l'aspect procédural du processus de reformulation et par conséquent au second des deux sous-systèmes proposés. Il est clair que les deux sous-systèmes sont fortement dépendants : les règles procédurales doivent être capables d'activer les règles linguistiques et, symétriquement, les règles linguistiques doivent pouvoir être activées par les règles procédurales. Or, les deux sous-systèmes sont conçus pour des tâches distinctes et sont par conséquent soumis à des contraintes différentes, visant à une bonne adéquation des outils à leurs tâches respectives. Le sous-système linguistique tendra à proposer des règles riches permettant de décrire simplement la multiplicité des phénomènes linguistiques. Ces règles nécessitent pour leur activation des règles procédurales complexes, à la mesure de la richesse des règles linguistiques. Le sous-système procédural tendra, lui, à proposer des règles assez simples pour que leurs propriétés mathématiques et calculatoires soient bien connues, bridant par conséquence l'expressivité des règles linguistiques.

Les exigences émanant des deux sous-systèmes étant dans une certaine mesure contradictoires, nous allons être confrontés à un problème de choix : celui de décider quelles exigences satisfaire en priorité. Deux attitudes extrêmes peuvent être envisagées face à ce choix : la première consiste à ne pas se soucier de l'activation de la *TST* et de ne s'intéresser qu'au premier des deux sous-modèles distingués par I. Mel'čuk. Le second sous-système n'est alors conçu qu'en fonction des besoins qu'exprimera le premier. C'est ce que l'on pourrait appeler l'attitude du linguiste, elle est assez proche de celle adoptée par I. Mel'čuk, comme le laissent apparaître les trois principes suivants, extraits des *cinq principes fondamentaux de la TST* proposés dans [Mel'čuk, 1988b] :

- La modélisation cybernétique de l'intuition linguistique native plutôt que l'étude mathématique des systèmes formels représentant le langage.

- Les formalismes doivent servir le linguiste et non le contraire.

- Différents langages formels pour différents niveaux de représentation plutôt que la quête d'une représentation homogène.

A l'autre extrémité, on cherchera à satisfaire en priorité les exigences du second sous-système qui imposera ses contraintes au premier, ce que l'on pourrait appeler l'attitude de l'informaticien.

Notre position particulière nous interdit la première attitude : en effet, les contraintes de mise en œuvre informatique que nous nous sommes imposées sont clairement en contradiction avec celle-ci. Afin de clarifier notre attitude vis-à-vis de ces deux extrêmes, nous allons décrire plus en détail ces contraintes de mise en œuvre informatique afin de mieux cerner leurs conséquences sur les règles linguistiques :

- Degré de formalisation du langage de représentation : les descriptions linguistiques représentées dans le système sont destinées à être interpretées par un ordinateur. Elles doivent donc être précises et ne doivent pas faire appel à l'intuition et au bon sens humains pour être correctement interpretées.

- Economie des outils de représentation : différents langages de représentation supposent différents mécanismes d'interprétation. Nous chercherons, dans la mesure du raisonnable, à restreindre le recours à des langages de représentation différents aux seuls cas où certaines informations ne peuvent être représentées dans un langage unique. Ce principe d'économie induit une homogénéité des moyens de représentation des connaissances, conséquence en contradiction avec le troisième principe de la *TST* énoncé ci-dessus, illustrant clairement les différences des deux approches.

- Organisation des connaissances linguistiques favorisant l'efficacité des traitements : la recherche d'efficacité des traitements informatiques nous poussera à proposer un regroupement des connaissances sensiblement différent de celui que propose la *TST*. Ce regroupement affectera principalement la répartition des connaissances entre composantes de la *TST* et entre ces dernières et le *DEC*.

- Séparation des connaissances linguistiques et des modules de traitements : nous avons tenu à faire la distinction, dans les trois buts que nous nous sommes assignés, entre les modules chargés de réaliser les phases du processus de génération et les connaissances auxquelles ils font appel. L'idée est d'adopter une approche déclarative et de représenter, autant que possible, les connaissances indépendamment des procédures réalisant les phases du processus, ceci afin de permettre l'évolution des connaissances linguistiques du système sans en modifier les procédures. On peut rapprocher cette distinction de la division effectuée dans la *TST* entre règles linguistiques et règles procédurales bien que dans notre cas, les connaissances linguistiques soient explicitement représentées en fonction des mécanismes procéduraux.

Au vu de ces exigences, le risque de tomber dans les excès de la seconde approche, celle de l'informaticien, est réel. Nous tenterons tout au long de cette thèse d'éviter cet ecueil et ferons en sorte que les exigences procédurales n'imposent jamais de contraintes inacceptables sur les règles linguistiques.

C'est donc une attitude assez libre que nous avons choisi d'adopter vis-à-vis de la *TST*. Nous aurions pu choisir d'utiliser les règles linguistiques mises à notre disposition par la *TST*, quitte à proposer un ensemble de règles procédurales complexes et peu efficaces. Cette attitude était trop en contradiction avec notre but premier qui est la mise en oeuvre effective du processus de reformulation. Nous avons préféré rechercher un compromis entre les deux attitudes extrêmes décrites ci-dessus, quitte à s'éloigner souvent et substantiellement de la *TST*, comme nous le verrons dès la section suivante.

3.2 Arbres élémentaires

La recherche d'une plus grande efficacité des traitements et la prise en compte des règles d'écriture de langues contrôlées vont nous amener à modifier sensiblement l'organisation des informations proposée dans la *TST*. Nous allons introduire et justifier cette réorganisation en envisageant deux étapes de notre processus que sont la construction des *SSyntS* d'une phrase et l'application des règles de paraphrase, dans le cadre de la *TST* ≪ pure ≫ ; en respectant d'une part ses niveaux de représentation et d'autre part les règles permettant d'effectuer les transitions entre niveaux de représentation adjacents. L'objectif recherché est de montrer que les regroupements proposés ne permettent pas toujours un traitement compatible avec les exigences d'efficacité que nous nous sommes fixées en 3.1. Nous introduirons alors un mode alternatif de représentation des connaissances, sous forme d'*arbres élémentaires*, et de relations entre arbres élémentaires permettant un traitement automatique plus efficace. Les arbres élémentaires tel que nous les proposons s'inspirent d'une part des grammaires d'arbres adjoints lexicalisés (LTAG) définies dans le cadre des grammaires syntagmatiques et d'autre part des *dendrogrammaires* développées dans

[Gladkij & Mel'čuk, 1975] en relation avec la *TST*. Nous palerons des premières en 4.1.4 et des secondes en 3.5.

3.2.1 Construction de la *SSyntS* d'une phrase et arbres élémentaires de surface

Dans le cadre de la *TST*, la *transition syntaxique de surface*, que ce soit dans le sens de la génération (*SSyntS* $\longrightarrow$*RMorphP*) ou dans celui de l'analyse (*RMorphP* $\longrightarrow$*SSyntS*) est effectuée grâce aux informations contenues dans le *composant syntaxique de surface*. Celui-ci est responsable des quatre opérations principales suivantes :

- La morphologisation
- La linéarisation
- Les phénomènes d'ellipses
- La prosodisation.[2]

En accord avec les limites que nous avons établies au chapitre 2, nous ne nous intéresserons qu'aux deux premières opérations. On pourra remarquer que celles-ci sont envisagées, du moins au niveau terminologique, dans le sens de la génération. Dans le sens de l'analyse, la construction de la *SSyntS* à partir de la *RMorphP* revient principalement à expliciter les dépendances de surface existant entre les éléments de la *RMorphP*, opération que nous appelerons hiérarchisation de la *RMorphP*, et qui est en fait l'opération inverse de la linéarisation. La *TST* propose trois types de règles pour mener à bien la morphologisation et la linéarisation :

1. Les règles syntagmes

   Elles constituent l'outil de base de la composante syntaxique de surface. Elles établissent une correspondance entre une dépendance syntaxique de surface et sa réalisation au niveau morphologique profond, en terme des positions relatives du gouverneur et du dépendant dans la chaîne linéaire et des éventuels marqueurs morphologiques qui leur sont associés.

   Une règle syntagme est une expression de la forme :

   $$A \Longleftrightarrow S \,|\, C$$

   Où A est un sous-arbre syntaxique de surface comportant au moins une dépendance, S une séquence ordonnée des noeuds de A et C une liste de conditions d'applicabilité de la règle. La règle sujet extraite de [Mel'čuk, 1988a] est donnée en exemple dans la figure 3.1.

[2]Pour le texte écrit, la prosodisation est représentée par la ponctuation.

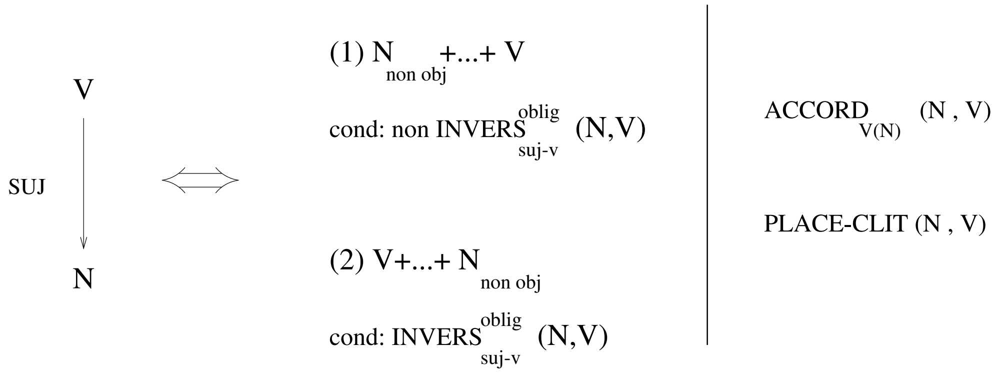


FIG. 3.1: Un exemple de règle syntagme

Cette règle stipule que, pour réaliser une construction sujet avec un syntagme nominal comme sujet, on doit mettre le sujet soit avant le verbe (la sous-règle (1) : seulement si la fonction standard« inversion obligatoire du sujet et du verbe» ne s'applique pas), soit après le verbe (la sous-règle (2) : seulement si la fonction standard« inversion obligatoire du sujet et du verbe» s'applique) ; le sujet ne doit pas être à la forme objective (cette exigence n'est pertinente que pour le pronom : me *vs* je, le/lui *vs* il, que *vs* qui) ; le verbe doit s'accorder avec le sujet (la fonction standard ACCORD V(N)) ; et le sujet manifesté par un pronom clitique ne doit pas être séparé du verbe si ce n'est par d'autres clitiques (la fonction standard PLACE-CLIT).

Dans le sens de l'analyse, les règles doivent être envisagées de droite à gauche : elles indiquent l'ordre linéaire et les marqueurs morphologiques qui trahissent la présence d'une dépendance syntaxique entre deux mots d'une séquence linéaire.

2. Règles d'ordonnancement des mots dans les syntagmes

   Les règles syntagmes ordonnent linéairement deux mots liés par une dépendance de surface, tels que le sujet et le verbe dans l'exemple précédent. L'ordre des mots partageant un gouverneur commun est, lui, régi par les règles d'ordonnancement des mots dans les syntagmes. C'est une règle de ce type qui, par exemple, contraint l'ordre entre le déterminant et l'épithète dans le groupe nominal *le grand bateau.* La représentation de telles règles n'a pas atteint dans les travaux de I. Mel'čŭk le niveau de formalisation des règles syntagmes : il n'est en particulier pas proposé de langage de représentation permettant de les exprimer. On pourrait les représenter, à l'instar des règles syntagmes, sous la forme d'une relation entre un sous-arbre et une suite ordonnée des noeuds du sous-arbre, comme dans la figure 3.2.

   Dans le sens de la génération, les règles d'ordonnancement représentent une composante importante de l'opération de linéarisation. Elles permettent d'ordonner entre eux les

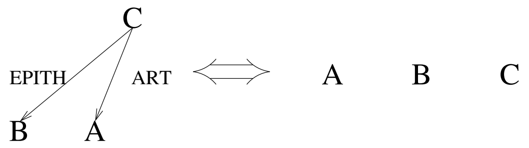


FIG. 3.2: Représentation graphique d'une règle d'ordonnancement

dépendants d'un même noeud pour obtenir un segment de phrase bien formé. La tâche équivalente dans le sens de l'analyse est d'interdire l'analyse d'une phrase présentant un ordre agrammatical. En effet, dans le cas de phrases agrammaticales, les règles d'ordonnancement ne peuvent s'appliquer, bloquant ainsi la hiérarchisation. De plus, règles d'ordonnancement des mots dans les syntagmes empêchent l'établissement de dépendances erronées lors de la hiérarchisation. Ainsi, dans le groupe nominal *la pompe du circuit secondaire*, c'est une règle d'ordonnancement interdisant la présence d'une épithète séparée du nom par un complément prépositionnel de ce dernier qui empêchera l'établissement d'une dépendance entre le nom *pompe* et l'adjectif *secondaire*.

3. Les règles d'ordre global

   Contrairement aux deux types de règles précédents, dont la fonction est clairement délimitée, le rôle des règles d'ordre global, qui prennent en compte les aspects de la linéarisation négligés par les règles précédentes, n'est pas clairement défini. Elles ≪ *déterminent le meilleur ordre possible étant donné une SSyntS, sur la base de données variées, telles que les indications contenues dans les règles syntagmes, les divisions donné-nouveau, topic-comment, les propriétés spécifiques à certains lexèmes.* ≫

Le processus de hiérarchisation tel que nous l'envisageons fait appel aux règles syntagmes et aux règles d'ordonnancement des syntagmes. Nous allons montrer dans ce qui suit que ces informations sont d'une part insuffisantes pour n'établir que les dépendances correctes entre les mots de la phrase, et sont d'autre part formulées sous une forme qui n'est pas optimale pour le traitement automatique.

La construction de la *SSyntS*, en ne tenant compte que des connaissances représentées au niveau du composant syntaxique de surface, peut aboutir à l'établissement de dépendances erronées. Un tel cas peut être illustré par la différence entre l'analyse des deux segments de phrases *donner ... à* et *manger ... à*. Dans le premier cas, la dépendance entre le verbe et la préposition peut être de nature actancielle ou circonstancielle, alors que dans le second cas elle ne pourra qu'être circonstancielle. Cette différence n'est pas prévisible dans une optique purement syntaxique, et l'application des règles syntagmes aux deux segments de phrases proposera pour les deux cas une lecture actancielle et une lecture circonstancielle. Dans le cadre de la *TST*, la mauvaise interprétation sera éliminée lors du passage en syntaxe profonde, lorsque les schémas de régime des mots de la phrase seront pris en compte. On saura alors que seul le verbe *donner* admet un complement d'objet indirect introduit par la préposition *à*.

Il serait donc intéressant, dans notre cas, de prendre en considération simultanément les informations syntaxiques provenant des règles syntagme ainsi que des aspects lexico-syntaxiques représentés au niveau du *DEC*, dans les schémas de régime des lexèmes, afin d'empêcher l'établissement de dépendances erronées et de ne pas poursuivre une voie que l'on sait vouée à l'echec. Une telle prise en compte de connaissances lexicales au niveau syntaxique de surface est envisageable sans modifier l'organisation de la *TST*. On pourrait imaginer de vérifier, lors de l'établissement de chaque dépendance de surface, que cette dernière n'est pas en contradiction avec le schéma de régime du gouverneur. Mais cette vérification ne peut être effectuée directement, car les indications contenues dans le schéma de régime du gouverneur ne sont pas formulées en termes de dépendances de surface comme le sont les règles syntagme, mais en terme de dépendances profondes. La prise en compte des contraintes du schéma de régime exige une étape supplémentaire de traduction de dépendances de surface en dépendances profondes. Une telle vérification, même si elle est possible en théorie, nécessite la mise en place de procédures relativement complexes qui sont en contradiction avec notre approche.

La prise en compte des règles d'ordonnancement des syntagmes pose un problème similaire. Faut-il les prendre en compte après application des règles syntagme et par conséquent invalider des dépendances qui auraient été établies par les règles syntagmes ? Là aussi, il serait préférable, pour des raisons d'efficacité, de les prendre en compte lors de l'établissement même des dépendances.

La solution que nous proposons à ces problèmes d'efficacité consiste à rassembler au sein de structures complexes appelées *arbres élémentaires de surface* (notés *AES*) les connaissances représentées dans les règles syntagme, dans le schéma de régime des entrées lexicales, ainsi que dans les règles d'ordonnancement des syntagme. Les arbres élémentaires de surface sont composés de dépendances syntaxiques de surface et de noeuds pouvant être plus ou moins contraints morphologiquement, lexicalement ou sémantiquement. Ils sont directement associés à des entrées lexicales et contraignent ainsi l'environnement morphologique, syntaxique, lexical et sémantique dans lequel une occurrence d'une entrée lexicale peut apparaître. L'analyse d'une phrase (construction des *SSyntS* lui correspondant) consiste alors à combiner entre eux les *AES* associés aux mots de la phrase. La structure et l'étendue des arbres *AES* seront discutées en détail au chapitre 4. Notons simplement que ce sont des considérations propres à la tâche d'analyse (construction de la *SSyntS*) qui déterminent la structure des *AES*.

Il convient de faire la différence entre deux types d'entités que nous allons manipuler. Nous trouverons d'une part des *AES* associés à des entrées lexicales, comme il apparaît dans la partie gauche de la figure 3.3. Dans de tels arbres, l'entrée lexicale, ici *donner*, constitue ce que l'on appellera l'*ancre lexicale* (terme emprunté aux LTAG) de l'*AES*. D'autre part, nous attribuerons à tout *AES* un *schéma* (noté $\mathcal{AES}$) qui peut être vu comme un *AES* sous-spécifié, auquel il manque une ancre lexicale, à l'image de l'arbre de droite de la figure 3.3. A chaque $\mathcal{AES}$ sera associé un identifieur unique qui constituera le *type* des arbres élémentaires conformes à l'$\mathcal{AES}$. Le schéma d'arbre de la figure 3.3 sera identifié par son nom : $vn_1n_2\grave{a}n_3$ et un arbre élémentaire de type S ayant pour ancre lexicale le lexème lex sera représenté par l'expression : $S(lex)$. L'*AES* de la figure 3.3, par exemple, sera représenté par l'expression $vn_1n_2\grave{a}n_3(donner)$. Dans la suite nous employerons indifféremment les termes schéma d'arbre élémentaire et type d'arbre élémentaire.

FIG. 3.3: Un arbre élémentaire de surface et son schéma

Dans un souci de lisibilité, nous n'avons pas représenté dans les arbres élémentaires de la figure 3.3 les différentes contraintes (morphologiques, lexicales, syntaxiques ou sémantiques) pouvant affecter les noeuds des arbres. Les noeuds sont représentés symboliquement par une catégorie ou un lexème. Il faut voir derrière ces catégories ou lexèmes une structure complexe de traits, comme nous le verrons en 4.1.7.

La mise en correspondance d'items lexicaux et de structures syntaxiques relativement complexes est assez courante dans les formalismes basés sur des grammaires de dépendances ([Hellwig, 1986a], [Fraser, 1989], [Sleator & Temperley, 1991] et [Gaifman, 1965]) ou en grammaires de constituants dans les arbres élémentaires des grammaires d'arbres adjoints lexicalisées (LTAG), avec lesquels nous comparerons nos *AES*, dans le chapitre 4. Ces structures présentent les avantages que nous avons vus dans cette section pour la construction des *SSyntS* d'une phrase. Elles vont de plus jouer un rôle central dans le processus de reformulation, rôle qui sera décrit en 3.3.

3.2.2 Représentation des règles de paraphrase et arbres élémentaires profonds

Les règles de paraphrase ont été introduites en 2.1.4, elles sont constituées de règles lexicales et de règles syntaxiques. Les premières représentent des équivalences sémantiques formulées en termes de fonctions lexicales, les secondes effectuent les restructurations requises par les changements lexicaux provoqués par l'application des règles lexicales.

La représentation des règles de paraphrase que nous proposons reprend la notion d'arbre élémentaire introduite dans la section précédente : une règle de paraphrase est vue comme une relation entre deux arbres élémentaires « paraphrastiques », relation que l'on appellera *relation paraphrastique*. Nous avons représenté dans la figure 3.4 un exemple de relation paraphrastique entre deux arbres élémentaires, représentant respectivement une structure verbale simple et une structure à verbe support.

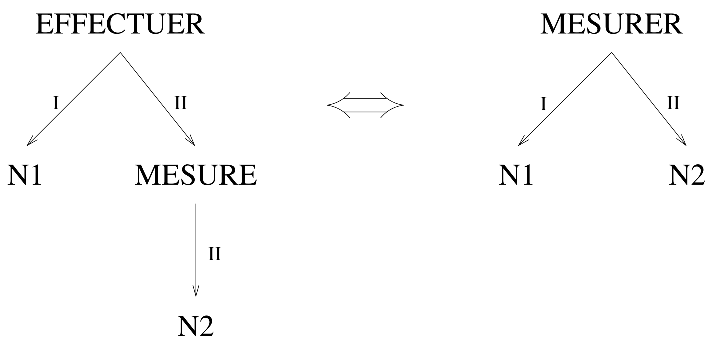


FIG. 3.4: Arbres élémentaires en relation paraphrastique

Les arbres élémentaires liés par une relation paraphrastique sont définis au niveau syntaxique profond, nous les appellerons donc *arbres élémentaires profonds* (notés *AEP*). Ils sont composés de dépendances profondes, de lexèmes profonds et de fonctions lexicales, se distinguant ainsi des *AES* introduits dans la section précédente. Graphiquement, les deux types de structures se distinguent par les étiquettes des dépendances (dépendances de surface dans un cas et dépendances profondes dans l'autre) ainsi que par une convention typographique, consistant à représenter en lettres majuscules les *lexèmes profonds* et en lettres minuscules les *lexèmes de surface*.

De la même façon que nous avions distingué dans la section précédente les *AES* et les types d'*AES* ($\mathcal{AES}$), nous distinguerons ici *AEP* et type d'*AEP* ($\mathcal{AEP}$). L'exemple de relation paraphrastique de la figure 3.4 met en jeu des *AEP*, alors que la règle de paraphrase qui a permi l'établissement de la relation paraphrastique est définie au niveau des $\mathcal{AEP}$. La règle permettant d'établir la relation entre les arbres de la figure 3.4 apparaît dans la figure 3.5. L'arbre de gauche représente la construction à verbe support ($Oper_1(N)$ est une fonction lexicale syntagmatique liant un nom N à un de ses verbes supports et $S_0(V)$ est une fonction lexicale paradigmatique liant un verbe V à certaines de ses nominalisations) et l'arbre de droite la construction à verbe sémantiquement plein (V)[3].

Cette représentation des règles de paraphrase peut être vue comme l'intégration d'une règle de paraphrase lexicale et des règles syntaxiques la desservant, au sein d'une structure plus étendue. La structure des *AEP* liés entre eux par relation de paraphrase est, par conséquent, déterminée par les règles de paraphrases lexicales et syntaxiques telles qu'elles sont définies dans la *TST*. Ce sont elles qui façonnent les *AEP*.

[3]Les conventions que nous avons adoptées pour les *AES* tiennent aussi pour les *AEP*. Les types d'*AEP* ou $\mathcal{AEP}$ seront identifiés par un nom. L'$\mathcal{AEP}$ apparaissant dans la partie droite de la figure 3.4, par exemple, aura pour nom $VN_IN_{II}N_{III}$. L'*AEP* associé au verbe *mesurer* sera représenté par l'expression $VN_IN_{II}N_{III}(MESURER)$. On pourra remarquer qu'un nom d'$\mathcal{AEP}$ s'écrit en majuscule, permettant ainsi de distinguer facilement $\mathcal{AES}$ et $\mathcal{AEP}$.

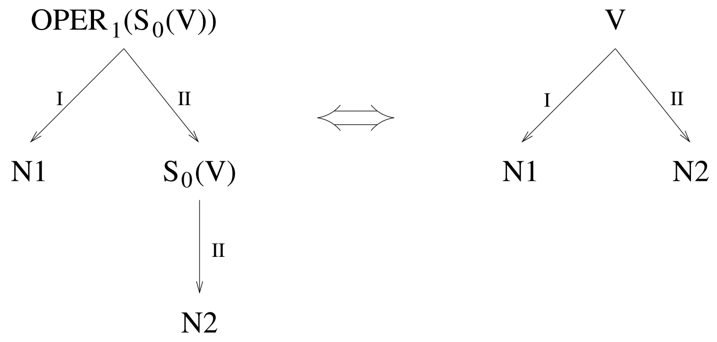


FIG. 3.5: Une règle de paraphrase de fission à verbe support

Cette intégration peut être rapprochée du regroupement de règles de natures diverses que nous avons effectué dans la section précédente pour donner naissance aux *AES*. On pourra remarquer que, de façon générale, les entités que nous allons manipuler tout au long du processus de reformulation se situent à un niveau de complexité supérieur à celui des entitées définies par la *TST* et peuvent être vues comme des regroupements de ces dernières.

3.2.3 Relations entre arbres élémentaires de surface et arbres élémentaires profonds

Nous avons introduit dans la section précédente un premier type de relation entre arbres élémentaires : la relation paraphrastique, liant deux *AEP*. Nous allons présenter ici un second type de relation reliant un *AEP* à un *AES*, permettant ainsi de faire le lien entre syntaxe de surface et syntaxe profonde. La relation entre *AES* et *AEP* lie un *AES* à son « équivalent » en syntaxe profonde. Nous appellerons cette relation *relation d'abstraction*, un *AEP* pouvant être vu comme une abstraction de plusieurs *AES* différents. On trouvera, figure 3.6, un *AES* et un *AEP* liés au verbe *donner*, reliés par relation d'abstraction. Le temps du verbe, représenté par une construction à deux nœuds (auxiliaire et participe passé) dans l'*AES*, est réduit à un trait dans l'*AEP*.

Les relations d'abstractions entre *AES* et *AEP* sont établies grâce à des *règles d'abstraction*. A l'instar des règles de paraphrase, les règles d'abstraction sont définies au niveau des types d'arbres élémentaires (ici entre un $\mathcal{AEP}$et un $\mathcal{AES}$) et s'appliquent entre arbres élémentaires (un *AEP* et un *AES*). La règle dont l'application est représentée dans la figure 3.6 est représentée dans la figure 3.7. Il est important de remarquer que l'$\mathcal{AEP}$ de la figure 3.7 est en relation d'abstraction avec plusieurs $\mathcal{AES}$ différents qui se distinguent par les prépositions introduisant leurs compléments, leur temps grammatical . . . Il y a donc, dans une grammaire complète, moins d'$\mathcal{AEP}$ que d'$\mathcal{AES}$.

Les relations établies entre arbres élémentaires (relation d'abstraction entre *AES* et *AEP* et relation de paraphrase entre *AEP*) permettent d'établir des liens entre les différents éléments de

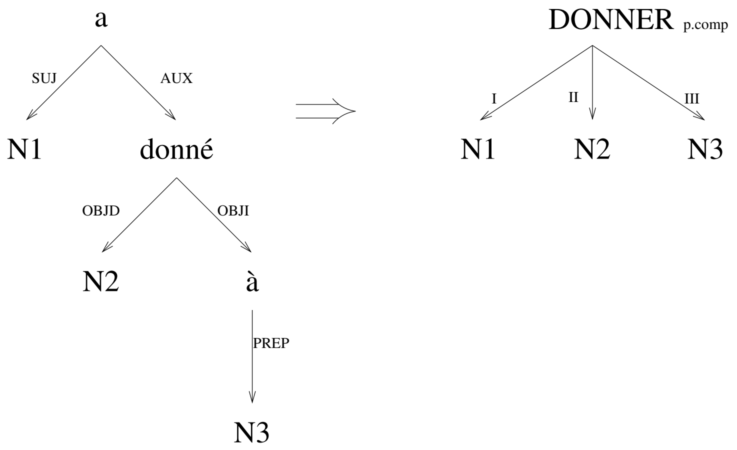


FIG. 3.6: Un *AES* lié à un *AEP* par relation d'abstraction

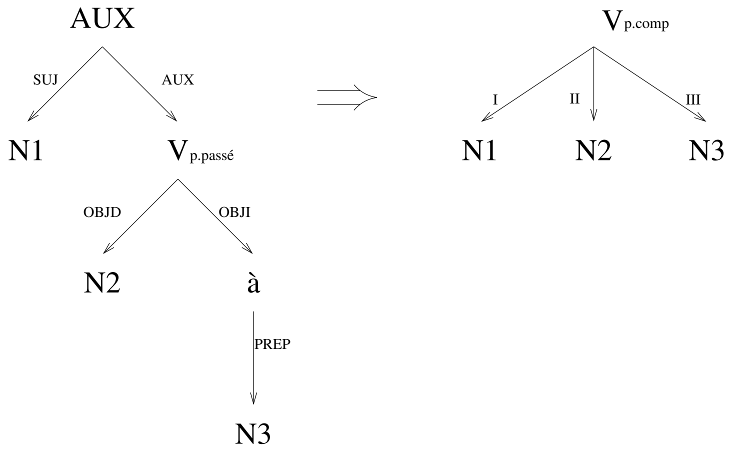


FIG. 3.7: Une règle d'abstraction

la grammaires, liens qui seront exploités lors des différentes phases du processus de reformulation. La phase d'analyse s'effectue par composition d'*AES* ; la transition syntaxique profonde est, elle, effectuée par substitution d'*AEP* aux *AES* leur correspondant par relation d'abstraction et la transformation de la *SSyntP* par substitution de certains *AEP* à d'autres *AEP* leur correspondant par relation de paraphrase. La phase de regénération d'une nouvelle phrase s'effectue par remplacement des *AEP* par les *AES* leur correspondant, puis par linéarisation de la structure ainsi obtenue. Ainsi, le processus entier se présente sous la forme de manipulation d'arbres élémentaires liés entre eux. La notion de relations entre arbres élémentaires se trouve par conséquent au cœur du processus de reformulation.

Fragilité de la relation d'abstraction

On pourra remarquer que les *AEP*, impliqués dans deux opérations différentes (la transition syntaxique profonde et la paraphrase), sont soumis à deux sources de contraintes distinctes. D'une part ils se définissent par rapport aux règles de paraphrase de la *TST* ; leur structure dépend alors des différentes règles de paraphrases. D'autre part, ils sont mis en correspondance par la relation d'abstraction avec des *AES*, subissant alors des contraintes structurelles de la part de ces derniers, dans la mesure où un *AES* doit correspondre structurellement à un *AEP*. Or la structure des *AES* est elle-même définie par des contraintes propres à l'analyse et nous verrons dans les chapitre 5 et 6 que ces deux sources de contraintes ne sont pas toujours compatibles : les *AEP* vus comme abstractions d'*AES* peuvent ne pas correspondre aux *AEP* vus comme éléments de base de la paraphrase. De façon générale, les règles de paraphrase pourront nécessiter l'existence d'*AEP* plus étendus que les *AEP* définis par abstraction d'*AES*. Nous serons amenés dans les chapitres 5 et 6 à revoir la définition des *AEP* ainsi que la nature des liens les *AEP* et les *AES* et les *AEP* entre eux.

3.3 Organisation de la grammaire et représentation des règles d'écriture d'une langue contrôlée

L'ensemble des structures présentées ci-dessus (*AES*, *AEP*, $\mathcal{AES}$ et $\mathcal{AEP}$) et les relations que nous avons établies entre elles (relation d'abstraction et relation de paraphrase) vont constituer la « grammaire » de notre système. Nous avons représenté dans la figure 3.8 l'ensemble de ces informations sous forme d'un tableau à trois colonnes. Dans chaque colonne ont été regroupées les structures de même nature : dans la colonne de gauche, les entrées lexicales, puis les $\mathcal{AES}$ et enfin les $\mathcal{AEP}$. Deux types de relations peuvent lier les différents éléments : des relations liant des entités de même nature et des relations liant des entités différentes. Dans la première catégorie, on trouvera les fonctions lexicales liant entre elles des entrées lexicales et les relations paraphrastiques liant des $\mathcal{AEP}$. Les relations liant une entrée lexicale à un ou plusieurs $\mathcal{AES}$ et un $\mathcal{AES}$ à un

$\mathcal{AEP}$ (relation d'abstraction) se situeront dans la seconde catégorie[4].

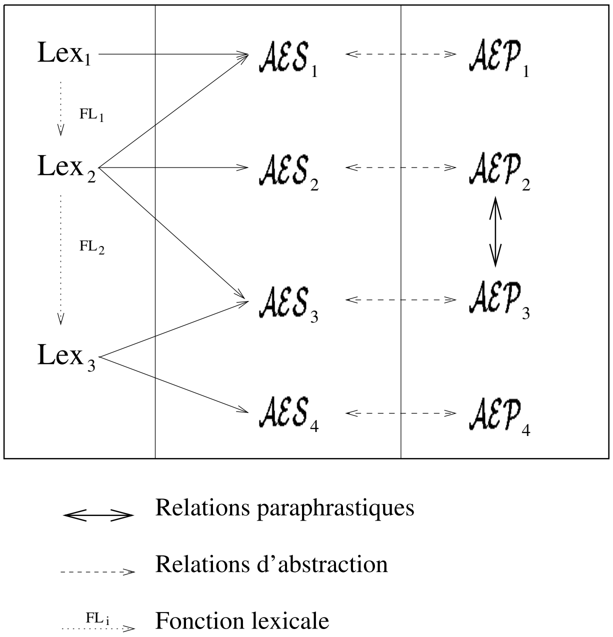


FIG. 3.8: Schéma d'organisation de la grammaire

Nous avons représenté dans la figure 3.9 une petite partie de la grammaire, composée des deux entrées lexicales *acheter* et *vendre*, liées entre elles par la fonction lexicale paradigmatique $Conv_{321}$ qui permet de relier un verbe à certains de ses converses. Chaque entrée lexicale est liée à des $\mathcal{AES}$ dans le lexique et ces derniers sont liés à des $\mathcal{AEP}$ par les règles d'abstraction. Les deux $\mathcal{AEP}$ sont reliés entre eux par une règle de paraphrase.

Cette organisation de la grammaire permet une représentation souple des règles d'écriture d'une langue contrôlée. Le principe consiste à marquer certains éléments de la grammaire comme interdits au regard de la langue contrôlée. Ces structures peuvent être de différentes sortes : lexèmes, *AES*, *AEP*, $\mathcal{AES}$ ou encore $\mathcal{AEP}$. C'est une représentation statique des règles d'écriture, elle n'explicite pas comment effectuer le passage d'une formulation interdite à une formulation autorisée, mais se contente de marquer les structures interdites dans la grammaire et le lexique.

[4]Nous ne nous sommes pas intéressé, par manque de temps, dans le cadre de ce travail, à la factorisation de la grammaire et du lexique : les $\mathcal{AES}$ et les $\mathcal{AEP}$ sont définis indépendamment les uns des autres bien qu'ils partagent souvent des parties ou des caractéristiques communes. C'est un travail nécessaire pour l'écriture d'une grammaire à large couverture, qui peut compter plusieurs dizaines d'$\mathcal{AES}$. Plusieurs aspects de la factorisation de la grammaire étudiés dans le cadre des *Word Grammars* par [Fraser & Hudson, 1992], dans le formalisme des *Head Driven Phrase Structure Grammars* (HPSG) par [Flickinger, 1987] et dans le cadre des grammaires d'arbres adjoints lexicalisées (LTAG)par [Candito, 1995] peuvent être mis à profit dans notre cadre.

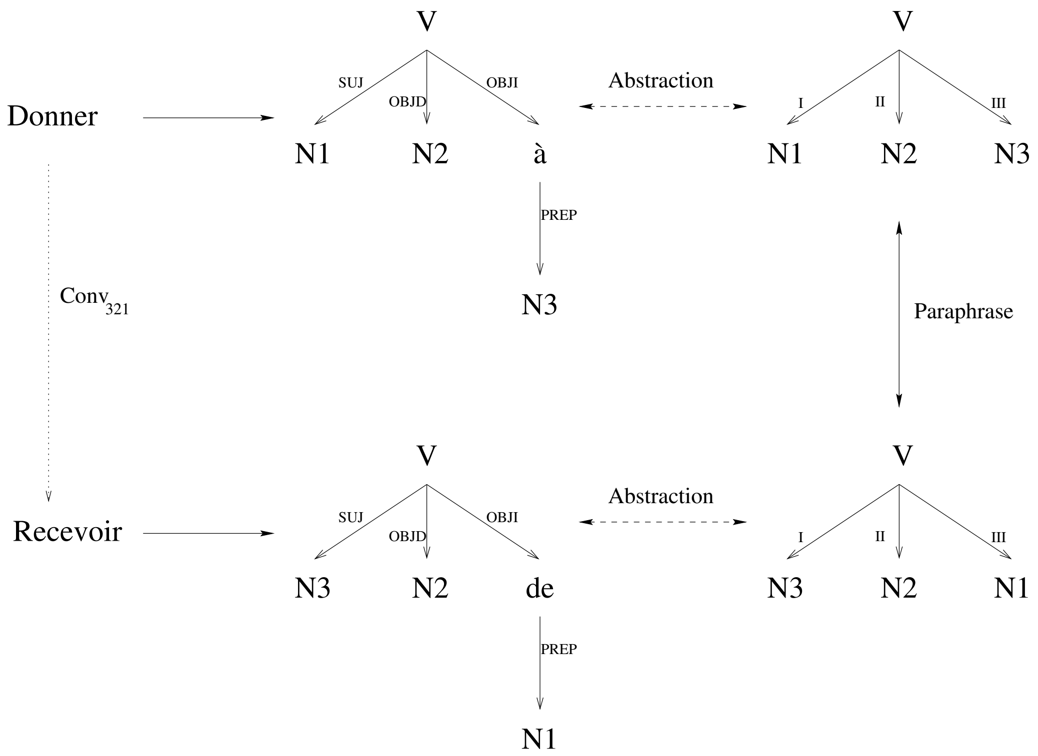


FIG. 3.9: Aperçu de la grammaire

L'avantage de cette représentation des règles d'écriture, rendue possible par l'existence de ces structures lexico-syntaxiques étendues que sont les arbres élémentaires, est que la détection des erreurs contenues dans une phrase se fera lors de l'analyse de la phrase. On saura en effet, à l'issue de l'analyse, si l'on a eu recours à des structures interdites pour construire la représentation syntaxique de la phrase. La phase de détection d'erreurs proprement dite est ainsi éliminée. De plus, cette représentation des règles d'écriture permet de résoudre le problème du partage des ressources linguistiques dans les systèmes de reformulation. Nous avions en effet remarqué au chapitre 1, que les systèmes de reformulation basés sur des architectures de systèmes de traduction automatique nécessitaient l'écriture de deux grammaires, une grammaire complète d'analyse et une grammaire de génération réduite aux structures syntaxiques autorisées par une langue contrôlée. Dans notre cas, une seule grammaire est suffisante. Cette dernière distingue toutefois les structures selon qu'elles sont autorisées ou interdites par la langue contrôlée.

3.3.1 Principes de marquage des structures interdites

Ce mode de représentation des règles d'écriture permet de marquer des« phénomènes» interdits de différentes natures. Ces derniers se divisent en deux grandes classes : les phénomènes syntaxiques indépendants de considérations lexicales et les phénomènes lexico-syntaxiques. Dans le premier cas, il s'agit de marquer un phénomène syntaxique représenté dans la grammaire par un type d'arbre élémentaire. Selon le type de phénomène à décrire, on marquera dans la grammaire un $\mathcal{AES}$ ou un $\mathcal{AEP}$. On peut ainsi interdire les constructions passives ou certaines constructions à verbe support en marquant les $\mathcal{AEP}$ correspondant à ces structures. On peut aussi interdire toutes les structures verbales trivalentes dont un des actants est introduit par la préposition *pour*, en marquant les $\mathcal{AES}$ correspondant à ces constructions. En ce qui concerne les interdictions lexicales, deux cas sont à distinguer : l'interdiction d'un lexème particulier, quelle que soit la structure syntaxique dans laquelle il apparaît, ou l'interdiction d'un lexème particulier dans une construction particulière. L'interdiction de l'usage du verbe *équiper*, par exemple, dans tout contexte syntaxique, entre dans le premier cas. Il sera réalisé par le marquage du lexème *équiper* dans le lexique. L'interdiction du même verbe, dans les constructions passives uniquement, se fera par marquage des *AES* passifs associés au verbe dans le lexique. Ce marquage revient, dans la figure 3.8, à marquer les liens qui relient une entrée lexicale à certains $\mathcal{AES}$ qu'elle peut ancrer. Le marquage d'un lien entre un lexème et un $\mathcal{AES}$ particuliers indique que l'*AES* obtenu par combinaison de ce lexème et de cet $\mathcal{AES}$ est interdit par la langue contrôlée. Le marquage d'arbres élémentaires dans le lexique permet ainsi d'interdire certains mots dans des constructions syntaxiques particulières sans pour autant interdire la construction syntaxique de façon absolue. Quatre cas d'interdictions peuvent donc être distingués :

- Interdiction d'un $\mathcal{AES}$.
- Interdiction d'un $\mathcal{AEP}$.
- Interdiction d'un *AES*, par marquage d'un lien entre un lexème et un $\mathcal{AES}$.
- Interdiction d'un lexème.

Cette représentation des règles d'écriture permet de dissocier les règles d'écriture propres à une langue contrôlée particulière, des connaissances linguistiques nécessaires à la reformulation, qui elles, sont propres au système de reformulation lui-même. C'est ce dernier qui possède les connaissances permettant de reformuler une phrase, mais ce sont les prescriptions du langage contrôlé qui indiquent quels phénomènes sont interdits, conformément à la dichotomie proposée dans le chapitre 1 entre connaissances linguistiques et prescriptions propres à une langue contrôlée. De plus, ce mode de représentation permet une mise à jour simple des règles d'écriture : enlever une règle revient à réhabiliter une structure interdite et ajouter une règle revient à interdire une structure précédemment autorisée. Les évolutions d'une langue contrôlée peuvent ainsi être prises en compte facilement.

3.4 Opérations de manipulation des arbres élémentaires

La mise en œuvre de la reformulation fait appel à deux opérations de manipulation d'arbres : l'attachement qui est une opération de composition d'*AES* et la substitution, qui est une opération de remplacement d'un arbre élémentaire par un autre au sein d'une structure plus étendue.

3.4.1 L'opération d'attachement

C'est l'opération qui permet de combiner deux arbres pour n'en former qu'un. Elle consiste à unifier la racine du premier arbre avec un noeud du second, appelé *site d'attachement*. L'opération d'attachement permet de combiner des *AES* entre eux lors de la phase de construction de la *SSyntS*, qui sera décrite au chapitre 5. Les détails de l'opération seront décrits au chapitre 4. Nous avons représenté dans la figure 3.10 deux exemples d'attachements : en haut l'attachement de deux *AES* et en bas, l'attachement de deux arbres de surface plus complexes.

3.4.2 L'opération de substitution

C'est l'opération qui permet de remplacer un arbre élémentaire par un autre qui lui est lié par une relation d'abstraction ou de paraphrase. La figure 3.11 illustre la substitution, dans une *SSyntP*, de l'*AEP* du verbe INSTALLER à l'*AEP* du verbe ÉQUIPER. Les arbres élémentaires substitués sont représentés en gras.

L'opération de substitution est utilisée dans trois cas :

1. Lors de la transition *SSyntS* $\longrightarrow$*SSyntP*, où elle permet de remplacer un *AES* par un *AEP* lui correspondant par règle d'abstraction.

2. Lors de la transformation de la *SSyntP* pour substituer à un *AEP* interdit, un *AEP* autorisé lui correspondant par relation paraphrastique.

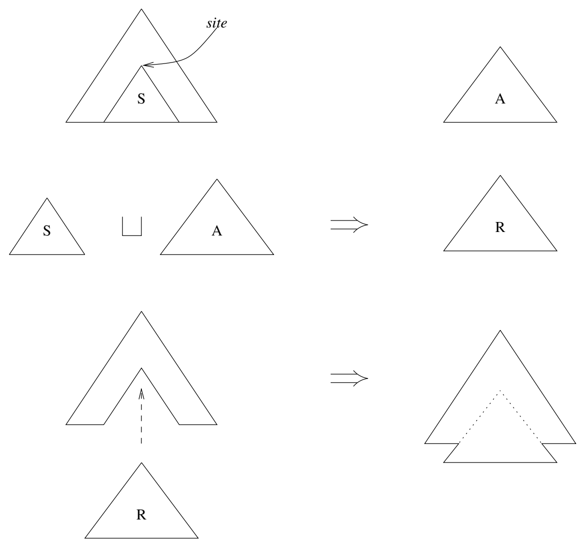


FIG. 3.10: Deux exemples d'attachement

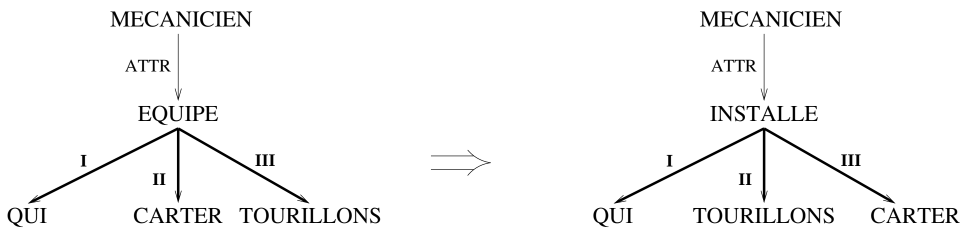


FIG. 3.11: Un exemple de substitution dans une *SSyntP*

3. Lors de la transition *SSyntP* $\longrightarrow$*SSyntS*, pour substituer des *AES* aux *AEP* entrant dans la composition de la *SSyntP*.

3.5 Les dendrogrammaires

Les notions d'*AES* et d'*AEP* ainsi que les relations définies entre elles peuvent être rapprochées des *dendrogrammaires* définies par [Gladkij & Mel'čuk, 1975]. Les dendrogrammaires constituent un système formel permettant l'implémentation de certaines transformations sur des arbres, en vue de mettre en œuvre la transition *SSyntP* $\longrightarrow$*SSyntS*, ainsi que la transformation *SSyntP* $\longrightarrow$*SSyntP* '. Les transformations sont regroupées au sein de grammaires manipulant des arbres, ou dendrogrammaires. Ces dernières sont composées d'une série de transformations élémentaires de la forme $(t_1 \Rightarrow t_2 \mid f)$ où t_1 et t_2 sont des arbres et f, une fonction partielle (que nous appellerons *fonction de correspondance*) des nœuds de t_1 sur les nœuds de t_2. Deux grandes familles de grammaires sont distinguées : les grammaires syntaxiques et les grammaires lexico-syntaxiques. Les premières manipulent des arbres syntaxiques purs, dont les nœuds ne sont pas étiquetés par des lexèmes, à l'image de nos $\mathcal{AES}$ et $\mathcal{AEP}$, alors que les secondes manipulent des arbres lexico-syntaxiques, à l'image de nos *AES* et *AEP*. Nos relations d'abstraction et de paraphrase entrent dans le cadre des transformations définies par les dendrogrammaires. De plus, deux opérations sont définies : une opération de composition permettant de combiner entre eux plusieurs arbres élémentaires et une opération d'application d'une transformation à un arbre.

Les dendrogrammaires n'ont pas connu, à notre connaissance, d'autres développements que l'article de A. Gladkij et I. Mel'čuk. Notre travail peut être vu comme une volonté de pousser plus avant ce mode de représentation. Notre apport consiste principalement à prendre en compte des contraintes linguistiques, inexistantes dans [Gladkij & Mel'čuk, 1975]. L'article de I. Mel'čuk et A. Gladkij reste en effet à un niveau de description très formel et ne décrit pas les principes linguistiques guidant la définition des arbres élémentaires. On ne sait pas, par exemple, quels types d'arbres élémentaires associer aux différentes catégories de discours, en particulier aux modifieurs. La prise en compte de contraintes linguistiques va enrichir le modèle en distinguant divers types d'arbres élémentaires, nécessitant des opérations de manipulations d'arbres élémentaires, tel que l'opération d'attachement, plus puissantes que les opérations définies dans les dendrogrammaires. Finalement, les relations entre arbres élémentaires vont être enrichies. Le modèle de représentation auquel nous allons aboutir à l'issue de cette thèse sera en fin de compte assez différent du modèle des dendrogrammaires. Les deux modèles partagent cependant une caractéristique commune importante qui est leur organisation autour de quatre éléments principaux : les *arbres élémentaires*, les *relations* entre arbres élémentaires établies grâce à des *règles* et, finalement des *opérations de manipulation* d'arbres élémentaires.

3.6 Les modules de transition

Cinq modules de transition permettent de mettre en œuvre le processus de reformulation. Les modules sont organisés séquentiellement, chacun prenant en entrée le résultat de l'étape précédente.

A l'exception du premier et du dernier, ces modules vont manipuler des arbres élémentaires à l'aide des opérations d'attachement et de substitution. Nous allons illustrer les différentes phases à l'aide d'un exemple : il s'agit de reformuler la phrase « *Le mécanicien équipe le carter de faux tourillons* », qui comporte le verbe *équiper*, interdit par le français rationalisé.

Analyse morphologique

Cette étape permet d'associer à tout mot de la phrase à traiter des *AES* ancrés par ce dernier. Elle se décompose en trois étapes :

- Analyse morphologique des mots de la phrase : les mots de la phrase sont décomposés en un ou plusieurs lexèmes enrichis de traits morphologiques.

- Détection des expressions composées : les séquences de lexèmes correspondant à des expressions composées répertoriées dans un dictionnaire des mots composés seront regroupées en une seule unité.

- Extraction des arbres élémentaires : pour chaque lexème ou expression composée munis de leurs traits morphologiques, les arbres élémentaires leur correspondant sont extraits du dictionnaire syntaxique.

Les *AES* associés à la phrase d'exemple ont été représentés dans la figure 3.12. A un mot sont généralement associés plusieurs *AES* décrivant plusieurs contextes possibles. Nous nous sommes limités ici à un *AES* par mot, pour des raisons de lisibilité. C'est bien entendu toujours le « bon » *AES* qui a été représenté. On remarquera que l'*AES* associé à la préposition régie *de* est réduit à sa plus simple expression, un noeud.

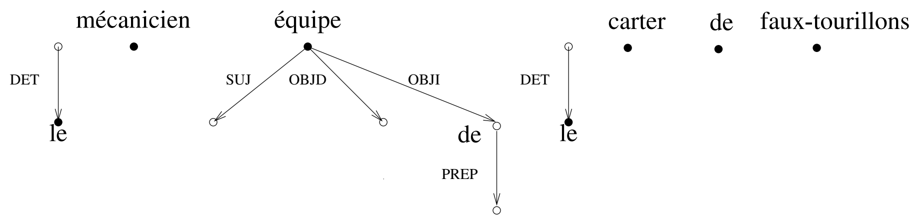


FIG. 3.12: Ensemble des *AES* des mots de la phrase

On remarquera aussi, dans la figure 3.12, que la juxtaposition des mots *faux* et *tourillons* a été reconnue comme mot composé : il est représenté par un seul *AES*.

Construction de la *SSyntS* d'une phrase

Elle consiste à combiner entre eux, grâce à l'opération d'attachement, les arbres élémentaires extraits du dictionnaire syntaxique lors de l'étape précédente. La composition des *AES* est réalisée par un analyseur syntaxique qui sera décrit en détail au chapitre 5.

Le résultat de l'analyse se compose de deux structures : les *SSyntS* de la phrase et, pour chacune de ces dernières, les *AES* qui ont servi à sa construction, ainsi que les opérations d'attachement permettant de construire la *SSyntS* à partir des *AES*. Cette structure sera représentée par un arbre dont les noeuds sont constitués d'arbres élémentaires. Les arcs représentent des opérations d'attachement : ils lient le noeud racine d'un *AES* à un noeud quelconque d'un autre *AES*, site d'attachement du premier. Nous appellerons cette structure la *structure syntaxique de surface analytique* ou *SSyntS Analytique* et nous réserverons l'appellation *SSyntS* à la première des deux structures. Le rapport qu'entretiennent ces deux structures peut être directement comparé au rapport entre arbre de dérivation et arbre dérivé, défini dans les grammaires d'arbres adjoints, sur lesquelles nous reviendrons en 4.1.4.

Nous avons représenté dans la figure 3.13 une *SSyntS* et une *SSyntS Analytique* de de la phrase d'exemple. Les arcs liant la racine des *AES* à leur site de rattachement sont représentés en pointillés.

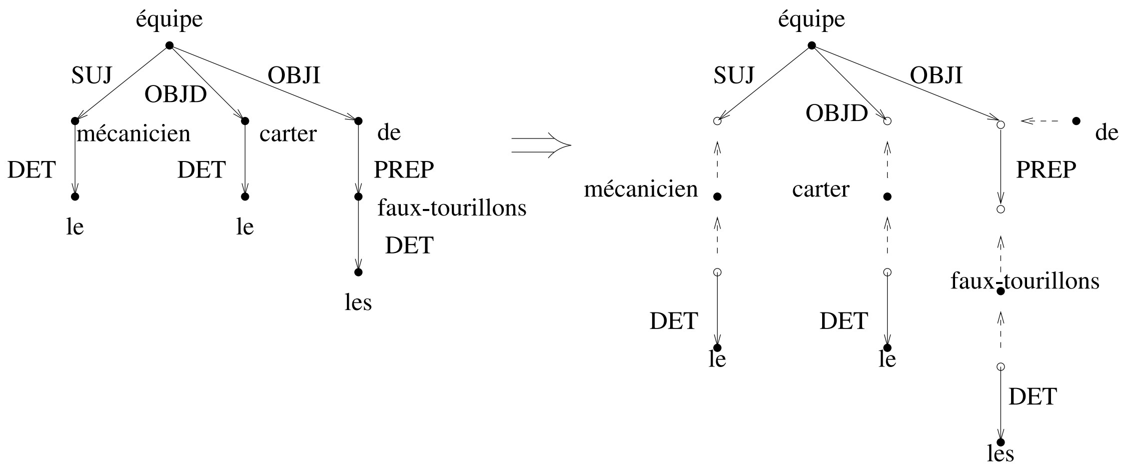


FIG. 3.13: Une *SSyntS* et une *SSyntS Analytique*

En un sens, la *SSyntS Analytique* est plus riche que la *SSyntS*, dans la mesure où la première permet de reconstruire la seconde, par réalisation des opérations d'attachement représentées par les liens en pointillé. Le contraire n'est par contre pas vrai : on ne peut dériver directement une *SSyntS Analytique* d'une *SSyntS*. Une *SSyntS Analytique* peut aussi être vue comme une *SSyntS* « partitionnée » en *AES*.

Construction de la *SSyntP*

C'est la *SSyntS Analytique* produite par l'analyseur lors de l'étape précédente qui est utilisée pour la construction de la *SSyntP*. Cette construction se fait par remplacement des *AES* de la *SSyntS Analytique* par les *AEP* leur correspondant au moyen d'une relation d'abstraction. A l'issue de cette opération, une structure syntaxique profonde analytique (*SSyntP Analytique*) est créée. Cette dernière permet de construire une *SSyntP*.

Cette phase sera décrite plus en détail au chapitre 5. La figure 3.14 représente la *SSyntP Analytique* construite à partir de la *SSyntS Analytique* de la figure 3.13.

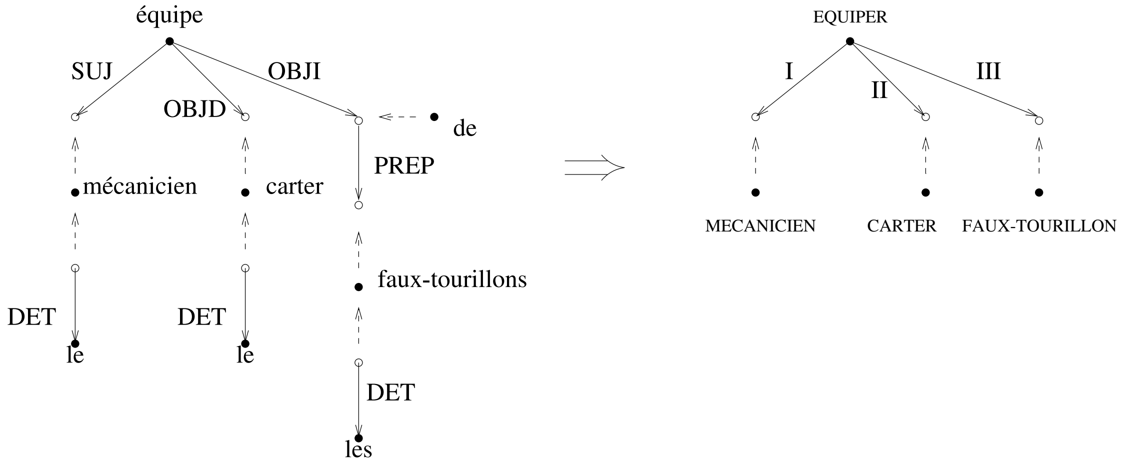


FIG. 3.14: Une *SSyntS Analytique* et une *SSyntP Analytique*

Dans la *SSyntP Analytique*, les déterminants des noms ne sont plus représentés explicitement, mais par des traits attachés aux noeuds de l'arbre. De même, la préposition *de* régie par le verbe *équiper* n'est plus représentée.

Transformation de la *SSyntP*

C'est la première phase de la reformulation proprement dite, elle consiste à remplacer, dans la *SSyntP Analytique* obtenue précédemment, des *AEP* interdits par des *AEP* autorisés leur correspondant par relation paraphrastique, donnant ainsi naissance à une nouvelle *SSyntP Analytique*. Cette phase se décompose en deux étapes : la recherche d'*AEP* ≪ autorisés ≫ liés par relation paraphrastique aux *AEP* ≪ interdits ≫ apparaissant dans la *SSyntP Analytique*, suivi du remplacement effectif des *AEP* interdits par des *AEP* autorisés. Lors de la première étape, plusieurs *AEP* peuvent être candidats à la substitution d'un *AEP* interdit : des critères devront être définis pour effectuer un choix. Cet aspect sera étudié au chapitre 6.

Dans l'exemple de la figure 3.15, l'*AEP équiper*, interdit dans le cadre du français rationalisé, est remplacé par l'*AEP* associé au verbe *installer*. A l'issue de cette phase, nous serons en présence d'une *SSyntP Analytique* composée exclusivement d'*AEP* autorisés.

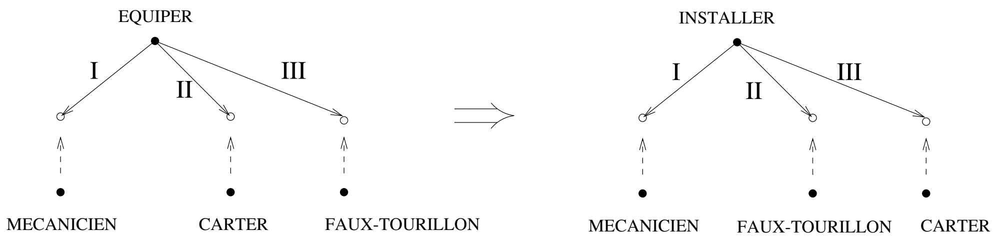


FIG. 3.15: Deux *SSyntP Analytiques*

On remarquera que, du fait que la substitution s'effectue au niveau syntaxique profond, certains détails, tels que la modification des prépositions introduisant les actants des verbes substitués, n'ont pas à être pris en compte explicitement : les bonnes prépositions seront introduites lors du passage en syntaxe de surface.

Reconstruction d'une *SSyntS*

La deuxième phase de la reformulation, consiste à remplacer les *AEP* de la *SSyntP Analytique* par des *AES* leur correspondant par relation d'abstraction. Elle constitue effectivement une phase de reformulation dans la mesure où à un *AEP* peuvent correspondre plusieurs *AES*, certains d'entre eux pouvant être≪ interdits≫. Il s'agit alors de ne choisir que des *AES*≪ autorisés≫. A l'issue de cette phase, une *SSyntS Analytique* constituée d'*AES*≪ autorisés≫ aura vu le jour. Elle permettra de reconstruire une *SSyntS*(voir figure 3.16).

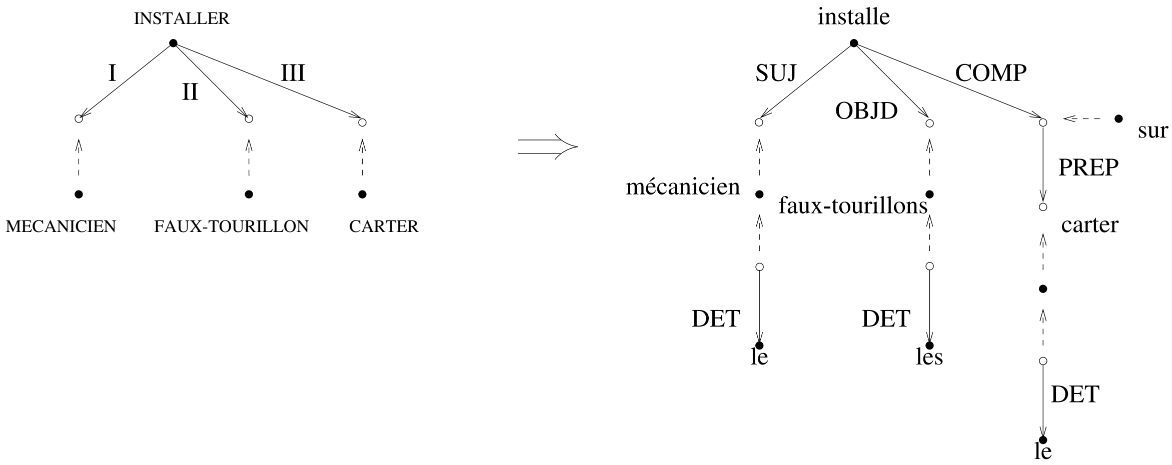


FIG. 3.16: Une *SSyntP Analytique* et une *SSyntS Analytique*

Linéarisation de la *SSyntS*

Troisième et dernière étape de la reformulation : elle consiste en la linéarisation et la morphologisation de la *SSyntS* obtenue lors de l'étape précédente. Elle constitue aussi une étape de la reformulation, dans la mesure où elle prend en compte des contraintes sur l'ordre des mots dans la phrase générée.

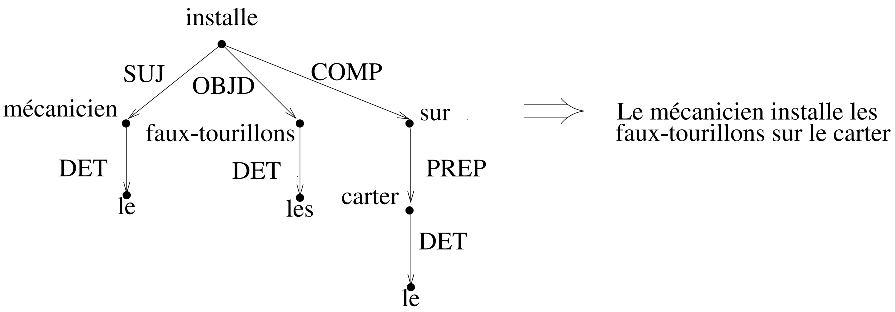


FIG. 3.17: Linéarisation d'une *SSyntS*

Chapitre 4

Les arbres élémentaires de surface et l'opération d'attachement

Nous avons proposé au chapitre précédent une modélisation des connaissances lexico-syntaxiques sous forme d'arbres élémentaires ainsi que des opérations de manipulation de ces derniers. Nous allons décrire en détail dans ce chapitre les deux notions d'arbres élémentaires de surface et d'attachement. Le traitement de faveur dont bénéficient ces deux notions se justifie en partie par leur complexité, les *AES* intègrent en effet de nombreuses informations provenant de plusieurs parties de la *TST*. Cette complexité des *AES* se reflète dans l'opération d'attachement. D'autre part, les *AES* et l'opération d'attachement constituent la partie de notre formalisme que l'on retrouve dans la plupart des formalismes grammaticaux générativistes aussi bien en grammaire de dépendances qu'en grammaires syntagmatiques. Nous profiterons de cette présentation détaillée pour comparer ces deux notions à leurs équivalents dans d'autres formalismes grammaticaux et notamment dans les grammaires d'arbres adjoints lexicalisées (LTAG) avec lesquelles notre formalisme présente de grandes similarités. Nous commencerons par décrire en détail la structure des arbres élémentaires, en nous penchant sur la manière dont elles intègrent des informations morphologiques, syntaxiques, lexicales et sémantique et sur leur domaine de localité. Nous proposerons ensuite des principe de bonne formation permettant de guider leur construction. Nous proposerons alors une représentation des arbres élémentaires à l'aide de structures de traits complexes. Le chapitre s'achèvera sur une description détaillée de l'opération d'attachement.

4.1 Les arbres élémentaires de surface

Nous avons mis en évidence, dans le chapitre 3, certains problèmes que pose l'organisation des informations au sein de la *TST* pour la mise en oeuvre automatique du processus de reformulation. Après analyse de ces problèmes, nous avons proposé la notion d'arbres élémentaires de surfaces (*AES*) comme moyen de représentation des informations linguistiques adapté à la tâche d'analyse et à la description de structures lexico-syntaxiques interdites par une langue contrôlée. Le principe qui sous-tend la notion d'*AES* est de contraindre au maximum le contexte d'occurrence d'un mot, dans le but de limiter autant que possible les cas de fausses ambiguïtés pouvant apparaître lors de l'analyse syntaxique, et d'optimiser ainsi les traitements automatiques. Cette section a pour but de décrire en détail les *AES*.

Nous commencerons par décrire le formalisme pour grammaires de dépendances de H. Gaifman qui présente, outre son interêt historique, l'avantage d'une grande simplicité. Nous identifierons dans ce formalisme un certain nombre de limites et nous verrons comment les *AES* permettent de les dépasser. Cette façon de procéder nous permettra de comparer nos *AES* à plusieurs autres formalismes pour grammaires de dépendances qui ont été proposés dans la littérature. Le but de cette section n'est pas de décrire en détail les différents formalismes que nous allons citer, mais plutôt de mettre en relief certaines de leurs caractéristiques qui nous semblent importantes et qui ont influencé notre réflexion.

4.1.1 Le formalisme de H. Gaifman

Les travaux décrits dans [Gaifman, 1965, Hays, 1964] et [Robinson, 1970] s'inscrivent dans un mouvement de formalisation des grammaires de dépendances. Le but de ces travaux est de proposer un système génératif, fondé sur les structures de dépendances et de comparer la puissance générative de ce type de grammaires à celle des grammaires de réécriture syntagmatiques. Nous ne nous intéresserons pas ici à la capacité générative du formalisme proposé, ni à la comparaison entre grammaires de dépendances et grammaires syntagmatiques, mais aux connaissances linguistiques représentées, ainsi qu'au formalisme de représentation. Bien que les travaux de D. Hays aient précédé ceux de H. Gaifman, c'est à ce dernier que l'on attribue généralement la paternité du formalisme.

Le formalisme est constitué de trois ensembles de règles :

1. Un ensemble fini de règles de la forme :

$$X(Y_1, Y_2, ..., Y_l, *, Y_{l+1}, ..., Y_n)$$

Où $Y_1, ..., Y_n$ et X représentent des catégories syntaxiques. Une telle règle indique que les catégories $Y_1, ..., Y_n$ peuvent dépendre de la catégorie X dans l'ordre indiqué dans la règle et que X occupe, par rapport à ses dépendants, la position matérialisée par un astérisque. Une telle règle peut être représentée graphiquement par un arbre de dépendances d'une profondeur égale à l'unité et ayant la catégorie X pour racine. Nous appellerons de tels arbres les *structures élémentaires* de la grammaire, et X la tête de la règle ou la racine de la structure élémentaire. Par la suite, nous emploierons indifféremment les termes règles ou structures élémentaires.

Nous avons représenté sous forme arborescente, dans la figure 4.1, la règle $V(N, *, Adv, N, Prep, Prep)$ indiquant qu'un verbe peut dominer un nom (son sujet) placé à sa gauche, un adverbe, un autre nom (son objet direct) et deux prépositions (d'autres compléments, actanciels ou circonstanciels, du verbe) placés à sa droite. Cette règle nous servira d'exemple dans la suite de cette section. L'ordre linéaire entre les catégories mises en jeu dans la règle est représenté par leur position horizontale dans l'arbre ; c'est ainsi, par exemple, que V est situé entre le premier N et Adv et que le second N est situé entre Adv et le premier $Prep$.

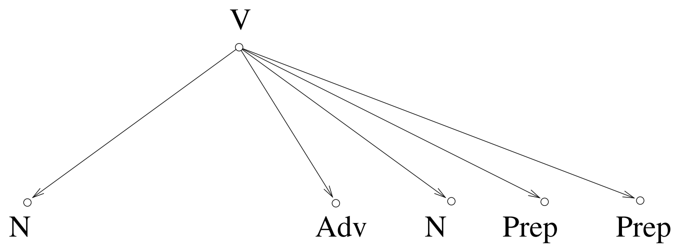


FIG. 4.1: Une règle sous forme arborescente

2. Un ensemble de règles donnant, pour chaque catégorie syntaxique, les mots appartenant à cette catégorie. En d'autre termes, le lexique.

3. Une règle donnant l'ensemble des catégories pouvant dominer une phrase, telle que la catégorie verbale V.

La représentation graphique des règles du premier ensemble, telle que la règle de la figure 4.1, n'est pas sans rapport avec les *AES* que nous avons présentés au chapitre précédent. Nous allons comparer ces deux types de structure indirectement, en indiquant dans les structures de H. Gaifman les cinq limites présentées ci-dessous, puis en décrivant la manière dont nous les avons dépassées dans les *AES*.

1. Insuffisance des critères de grammaticalité pris en compte dans les règles.

   A l'instar des règles de réécriture des grammaires syntagmatiques, les règles de H. Gaifman ne prennent en considération comme critère de grammaticalité que les catégories syntaxiques des mots et leurs positions relatives. Le lien entre le lexique et les règles est établi par le seul intermédiaire des catégories syntaxiques ; les autres caractéristiques des entrées lexicales ne sont pas représentées et ne sont pas prises en compte dans les règles syntaxiques. Or il est bien connu que ces informations sont nécessaires pour ne pas générer de phrases agrammaticales dans un cadre génératif et pour ne pas effectuer de faux rattachements dans un cadre d'analyse. Ainsi, notre règle d'exemple ne permet pas de représenter le phénomène d'accord entre le verbe et son sujet ni de contraindre la nature lexicale des prépositions introduisant les compléments du verbe.

2. Limites du domaine de localité des règles.

   Une règle représentée sous forme graphique se réduit à un arbre de profondeur égale à un. Cette limite peut se révéler insuffisante dans certains cas. Elle ne permet pas, par exemple, de représenter des contraintes sémantiques qu'un verbe peut imposer à ses dépendants indirects, il est impossible, en particulier, de représenter dans une règle associée au verbe *donner*, une contrainte sémantique sur l'objet indirect, car celui-ci n'est pas directement dominé par *donner*, il est, par conséquent, extérieur au domaine de localité de la règle.

3. Absence de représentation des fonctions syntaxiques.

   Les structures générées par les règles de H. Gaifman ne représentent pas explicitement les fonctions grammaticales des mots, indiquées en général par un étiquetage des dépendances. La règle d'exemple ne distingue pas les sujet, objet, circonstants et adverbes du verbe. Or, les fonctions grammaticales constituent une information importante pour les traitements que nous effectuerons et leur représentation explicite dans les structures syntaxiques est nécessaire.

4. Absence de distinction entre arguments et modifieurs.

   Une règle doit prévoir tous les dépendants possibles de sa tête, qu'ils soient obligatoires ou optionnels. Dans notre exemple, la règle prévoit aussi bien les sujet et objet du verbe que les prépositions introduisant des circonstants et d'éventuels adverbes. S'il semble légitime de représenter dans une règle associée à un verbe particulier les arguments de ce dernier, il est par contre moins naturel d'y représenter ses modifieurs possibles, que l'on retrouve dans la majorité des règles associées aux verbes. De plus, cette description exhaustive des dépendants de la tête de la règle aboutit à des règles très étendues.

5. Mode de représentation de l'ordre grammatical des mots dans les règles.

Les règles imposent un ordre total sur l'ensemble constitué des dépendants de la tête de la règle et de la tête elle-même. Ce mode de représentation ne permet pas de prendre en compte de façon naturelle la relative marge de liberté que les langues naturelles offrent généralement dans l'ordre grammatical des mots de la phrase. Ainsi, la règle d'exemple impose que l'objet direct du verbe apparaisse avant les deux compléments introduits par des prépositions. La prise en compte d'autres configurations, telle que l'insertion de l'objet direct entre les deux autres compléments, nécessite l'écriture d'une nouvelle règle :

$$V(N, *, Adv, Prep, N, Prep)$$

De plus, l'ensemble des phrases pouvant être engendrées par de telles règles se limite à la classe des phrases *projectives*, qui sera décrite en 4.1.5. Or la condition de projectivité est trop contraignante pour la langue naturelle, qui présente parfois des constructions non projectives.

Dans les trois sections suivantes, nous allons reprendre les inconvénients mentionnés ci-dessus et enrichir graduellement les règles de H. Gaifman pour aboutir aux *AES*.

4.1.2 Introduction de traits dans le lexique et dans les règles

Les règles syntaxiques de H. Gaifman ne prennent en considération que les catégories des mots et leur position relative comme critères de grammaticalité. Comme nous l'avions fait remarquer, ces deux informations sont insuffisantes pour interdire la génération de phrases agrammaticales, ne respectant pas, par exemple, l'accord entre le verbe et son sujet. Symétriquement, dans le sens de l'analyse, la prise en compte de ces seuls deux critères peut provoquer de faux rattachements.

Un premier enrichissement des règles, par l'introduction de *traits* au niveau des entrées lexicales et de conditions sur la valeur de ces traits dans les règles, à la manière de PATRII ([Shieber, 1986]), permet de prendre en compte de nouvelles connaissances linguistiques. Enrichie de la sorte, notre règle d'exemple se réécrit de la façon suivante :

$$V(N_0, *, Adv, Prep, N_1, Prep)$$
$$N_0. < num >= V. < num >$$

L'équation représentée sous la règle représente la contrainte d'accord entre le sujet et le verbe. Elle indique que le nombre du verbe (représenté par le trait $V. < num >$) doit être égal au nombre du sujet ($N_0. < num >$). L'introduction de traits au niveau des entrées lexicales et de conditions sur la valeur de ces derniers dans les règles permet de prendre en compte aussi bien des contraintes morphologiques, comme nous venons de le montrer, que des contraintes lexicales et sémantiques. Nous avions noté que la règle d'exemple ne permettait pas de contraindre la nature lexicale des prépositions introduisant les compléments du verbe, et aboutissait, dans certains cas, à la génération de structures agrammaticales. L'introduction des traits permet maintenant d'imposer ce type de contraintes, à l'image de l'exemple de la figure 4.2, où la nature lexicale de la préposition introduisant le premier complément du verbe est déterminée. Il est clair qu'une telle règle n'est adaptée qu'à certains verbes, qui partagent le même cadre de sous-catégorisation. Cet ensemble sera matérialisé par l'introduction d'un trait *sous-cat* au niveau des entrées lexicales et des règles, à la manière de GPSG ([Gazdar et al., 1985]). Enrichies de la sorte, les règles perdent en généricité pour gagner en précision et en richesse d'information. Ce mouvement de spécialisation des règles va s'accompagner d'une multiplication de ces dernières. Il y aura maintenant autant de règles associées aux verbes que de sous-catégories de verbes.

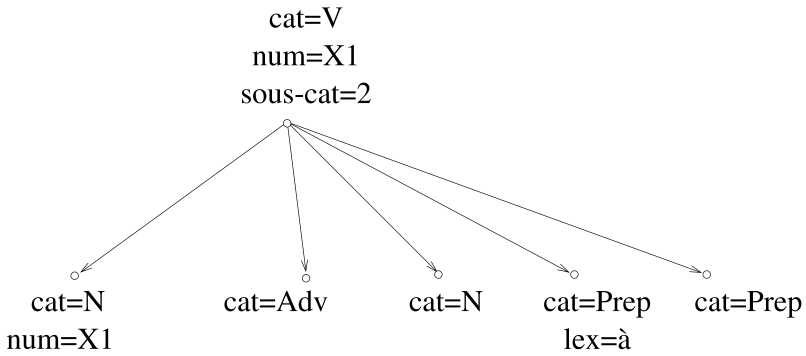


FIG. 4.2: Introduction de traits dans une structure élémentaire

On peut aller plus loin dans l'intégration des règles de syntaxe et du lexique en n'associant plus une entrée lexicale à une catégorie, et une catégorie à des règles comme c'était le cas jusque là, mais en associant directement à une entrée lexicale une ou plusieurs règles. La règle de la figure 4.2, par exemple, sera directement liée, dans le lexique, au verbe *donner*, faisant ainsi l'économie d'une représentation explicite du cadre de sous-catégorisation au niveau des entrées lexicales et de conditions sur ce dernier au niveau des règles. Ce lien direct entre les éléments du lexique et les structures élémentaires permet une autre vision des règles, qui peuvent maintenant être vues comme des *catégories complexes* qui décrivent explicitement le contexte syntaxique, lexical, morphologique et sémantique dans lequel peut apparaître une occurrence d'un mot. Ce dernier sera représenté dans la structure élémentaire par un nœud particulier, l'ancre lexicale de la structure élémentaire, introduite au chapitre précédent. Dans cette nouvelle perspective, le lexique associera aux entrées lexicales non pas une ou plusieurs catégories syntaxiques mais une ou plusieurs structures élémentaires que l'entrée lexicale peut « ancrer ».

La prise en compte d'informations lexicales dans la composante syntaxique d'une grammaire se retrouve dans plusieurs approches, aussi bien en grammaire de dépendances qu'en grammaire de constituants, dans un mouvement dit de *lexicalisation* de la grammaire qui, comme son nom l'indique, donne une place prépondérante aux informations lexicales. [Abeillé, 1991] défend l'idée d'une lexicalisation intégrale de la grammaire, tant du point de vue linguistique que du point de vue informatique. Bien que les arguments mis en avant le soient dans le cadre des grammaires de constituants, ils ne perdent rien de leur pertinence pour les grammaires de dépendances. Linguistiquement, la lexicalisation permet de mieux rendre compte, d'une part, de l'influence du contexte sur les propriétés des mots et, d'autre part, de l'influence des caractéristiques lexicales des mots sur les relations syntaxiques que peuvent entretenir ces derniers au sein de la phrase. La représentation de cette double influence est rendue possible par la représentation explicite du contexte au sein des *AES*.

En grammaire de constituants, l'approche qui a poussé le plus loin la lexicalisation de la grammaire est sans doute celle des grammaires d'arbres adjoints lexicalisées (*Lexicalised Tree Adjoining Grammars* ou LTAG [Schabes et al., 1988, Schabes, 1990]) qui associe à chaque entrée lexicale un ou plusieurs arbres syntagmatiques appelés arbres élémentaires. Nous reviendrons plus en détails en 4.1.4 sur ce type de grammaires.

Dans le cadre des grammaires de dépendances, la plupart des formalismes proposés dans la littérature s'inscrivent dans ce mouvement de lexicalisation. Les travaux présentés dans [Hellwig, 1986a, Jäppinen et al., 1986, Starosta & Nomura, 1986, Sleator & Temperley, 1991] et [Fraser, 1989] prônent tous une intégration du lexique et de la grammaire. Ils associent directement aux entrées lexicales une ou plusieurs structures élémentaires, prenant la forme d'arbres syntaxiques, au sein desquels l'entrée lexicale est représentée par un nœud particulier, l'équivalent de l'ancre lexicale.

Nous avons vu que dans le formalisme de H. Gaifman, les règles permettent de décrire les différents dépendants possibles de la tête de la règle, mettant ainsi l'accent sur la *valence active* des mots. C'est aussi le cas des travaux de [Hellwig, 1986a, Jäppinen et al., 1986] et [Starosta & Nomura, 1986]. Il peut également être intéressant dans certains cas de représenter, dans une structure élémentaire, la *valence passive* de l'ancre lexicale, ses gouverneurs possibles, afin d'imposer des contraintes sur ces derniers. On pourrait ainsi représenter, dans une structure élémentaire associée à un adjectif, des contraintes sur les noms que l'adjectif peut modifier. Cette possibilité est offerte dans les deux autres approches, celles de [Sleator & Temperley, 1991] et [Fraser, 1989], où sont représentées dans chaque structure élémentaire les valences actives et passives de l'ancre lexicale. On peut remarquer que dans ce dernier cas, toutes les dépendances seront représentées deux fois au niveau des structures élémentaires. La structure élémentaire associée à un article, par exemple, indiquera la valence passive de l'article (un nom). D'autre part, la structure associée au nom prévoira la place du déterminant dans sa valence active, aboutissant ainsi à une duplication systématique de toutes les dépendances dans la grammaire, situation peu économique du point de vue descriptif. Nous proposerons en 4.1.4 un moyen de représenter, dans les structures élémentaires, en fonction de la nature de l'ancre lexicale, soit la valence active de cette dernière, soit sa valence passive, soit les deux.

L'approche proposée dans [Genthial, 1991] se distingue des approches présentées ci-dessus en proposant une composante syntaxique plus « classique », basée sur des règles de réécriture. Ces dernières présentent dans leur partie gauche une liste ordonnée de catégories syntaxiques[1] et

[1] Il s'agit en fait d'une liste de sous-arbres dont la racine est constituée de ces catégories syntaxiques.

dans leur partie droite l'arbre construit à partir de cette liste. Le rattachement du nom et du déterminant, par exemple, sera traité par la règle suivante :

$$Det, Nom \rightarrow (Det)Nom$$

La structure parenthétique dans la partie droite représente la structure de l'arbre (le dépendant de Nom, Det, est représenté entre parenthèses). La règle dit que lorsqu'un déterminant précède directement un nom, il doit être rattaché à ce dernier.

Plus généralement, la règle suivante :

$$a, b, c, d \rightarrow a, (b)c, d$$

exprimera le fait que b se rattache à c dans le contexte a, d.

D. Genthial justifie l'existence d'une composante syntaxique autonome en remarquant que les grammaires lexicalisées (en particulier celles de [Hellwig, 1986a]), par l'amalgame qu'elles font entre la grammaire et le lexique, n'offrent pas de moyen de représenter des règles de grammaire d'une portée générale. Pour reprendre l'exemple du rattachement du déterminant, il est en effet plus satisfaisant de représenter ce phénomène par une règle générale stipulant qu'un article peut être rattaché à un nom, plutôt que de prévoir, dans toutes les structures élémentaires associées aux noms, la place du déterminant. Nous verrons en 4.1.4 la solution que nous proposons à ce problème de redondance des grammaires de dépendances lexicalisées.

4.1.3 Extension du domaine de localité des structures élémentaires

Il est intéressant, dans la perspective d'intégration du lexique et de la grammaire et de spécialisation des règles, de réexaminer le problème de l'étendue des règles que nous avions soulevé dans le second point.

S'il paraît difficile d'envisager l'extension du domaine de localité des règles à un niveau purement syntaxique, où rien ne peut être dit des caractéristiques lexicales des mots, la lexicalisation offre, par contre, un cadre naturel à une telle extension et permet de représenter différentes idiosyncrasies syntaxiques des mots dans les structures élémentaires. C'est pourquoi nous ne limiterons pas à un la profondeur des *AES*. Ces derniers pourront avoir une profondeur quelconque, comme dans l'exemple de la figure 4.3, représentant un *AES* de profondeur égale à deux.

On peut voir l'extension du domaine de localité des *AES* comme le regroupement de plusieurs règles (au sens de H. Gaifman) au sein d'une règle plus complexe, ce qui permet de contraindre l'utilisation de plusieurs règles simultanément. Nous verrons en 4.1.6 des principes de bonne formation des *AES* permettant en particulier de déterminer, pour une ancre particulière, l'étendue du domaine de localité de son (ses) *AES*.

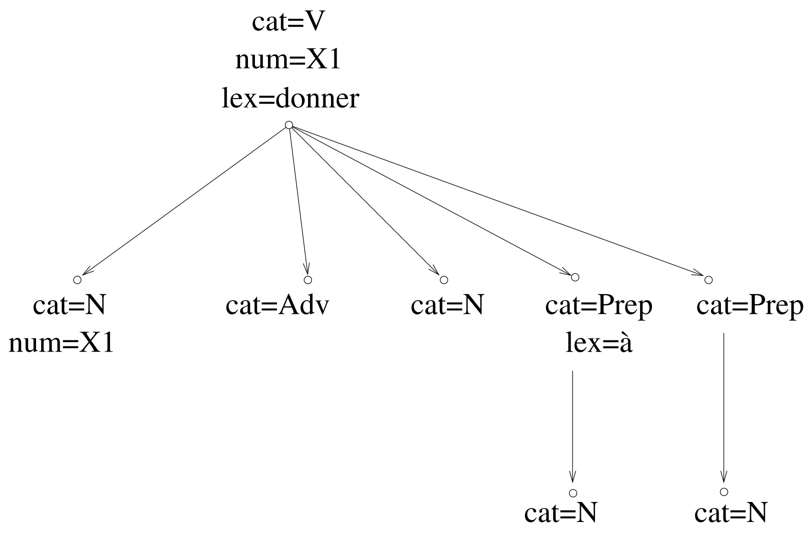


FIG. 4.3: Une structure élémentaire étendue

4.1.4 Typage des dépendances

En mettant l'accent sur les dépendances entre mots plutôt que sur la notion de constituant, les grammaires de dépendances offrent un moyen naturel de représenter les fonctions grammaticales des mots d'une phrase dans la représentation structurelle de cette dernière : les fonctions sont représentées grâce à un étiquetage des dépendances par la fonction syntaxique du mot situé à leur extrémité. Nous avons représenté dans la figure 4.4 la règle d'exemple enrichie de la représentation explicite des fonctions syntaxiques. Une telle représentation des fonctions grammaticales permet de mettre à jour des nuances qui n'étaient pas visibles dans la règle d'origine. Ainsi les deux prépositions dépendant du verbe, qui avaient le même statut dans la règle d'origine, se déclineront maintenant en une préposition introduisant un objet indirect (OBJI) et une préposition introduisant un complément circonstantiel (CIRC).

Nous ne rentrerons pas dans le détail des étiquettes des dépendances et des critères permettant de les définir : ce sont des aspects qui concernent davantage l'écriture d'une grammaire particulière. Du point de vue du formalisme, chaque dépendance de tout arbre élémentaire est définie par un nom et trois propriétés. Ces trois propriétés ne sont pas indépendantes ; on pourra trouver dans [Sgall, 1992] et [Lazard, 1994] des analyses de leur interdépendance. Nous n'aborderons pas ici ce problème et nous nous bornerons à énumérer les propriétés :

- répétable / non répétable

  Une dépendance de d est répétable s'il est possible d'avoir plus d'une dépendance étiquetée par d partant d'un même nœud. A titre d'exemple, une dépendance *épithète* sera généralement répétable, contrairement à la dépendance *sujet*.

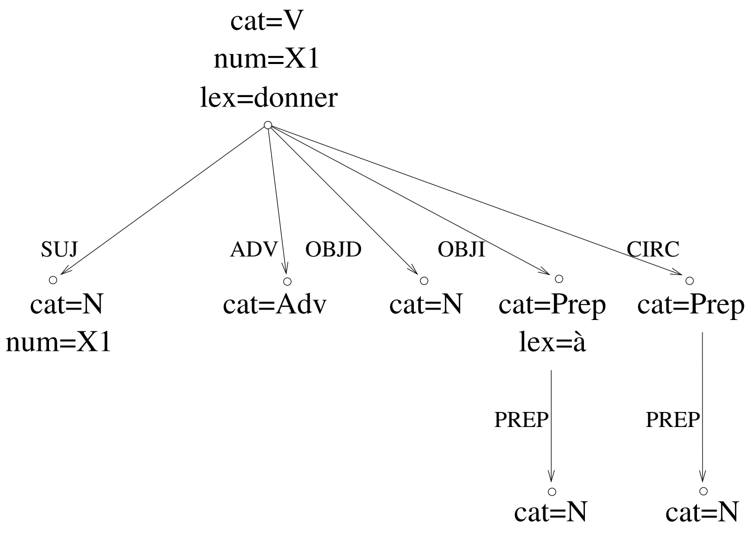


FIG. 4.4: Représentation des fonctions syntaxiques dans une structure élémentaire

- obligatoire / optionnelle

  Contrairement à la propriété de répétabilité, qui est définie au niveau de la définition d'une dépendance, la distinction obligatoire / optionnelle est définie au niveau d'une occurrence d'une dépendance au sein d'un arbre élémentaire particulier. Ainsi, la dépendance *objet direct* sera obligatoire dans l'arbre élémentaire associé au verbe *battre*, mais optionnelle dans l'arbre élémentaire de *manger*.

- argument / modifieur

  La propriété de répétabilité recoupe partiellement la distinction entre arguments et modifieurs. La frontière qui sépare les dépendances argumentales et modificatrices n'est pas toujours très claire et les critères permettant de la dessiner varient, lorsqu'ils existent, d'un auteur à l'autre. [Mel'čuk, 1988a] propose, par exemple, une conception assez étendue de la notion d'argument, rangeant dans cette dernière catégorie des dépendants que [Tesnière, 1959] ou [Lazard, 1994] identifieraient comme modificateur. De façon générale, les arguments sont plus liés à la nature du gouvernant que le sont les modifieurs. Dans le cas d'un verbe, le nombre et la nature des arguments qu'il peut régir sont des caractéristiques qui lui sont propres alors que ses modifieurs, même s'ils dépendent dans une certaine mesure du verbe lui-même, semblent moins contraints par ce dernier que ne le sont les arguments.

La prise en compte dans notre cadre de la distinction dépendances argumentales / dépendances modificatrices peut aboutir à une représentation plus économique de la grammaire. Nous avions remarqué en effet, et c'était un reproche que [Genthial, 1991] adressait aux grammaires de dépendances lexicalisées, que les structures élémentaires associées à un nom devaient prévoir tous les dépendants possibles de ce dernier, aboutissant à des règles très étendues et donnant l'impression d'un manque de factorisation. Ce problème peut être dépassé en ne représentant dans les arbres élémentaires associés à un lexème que les arguments de ce dernier. Les différents modifieurs potentiels constitueront des arbres élémentaires séparés, qui pourront être rattachés dynamiquement aux premiers grâce à l'opération d'attachement qui sera définie en 4.2. De cette façon, la règle d'exemple ne sera plus représentée par une seule structure élémentaire, mais, comme le montre la figure 4.5, par trois : la première associée au verbe, la seconde à l'adverbe et la troisième à la préposition introduisant le complément circonstanciel.

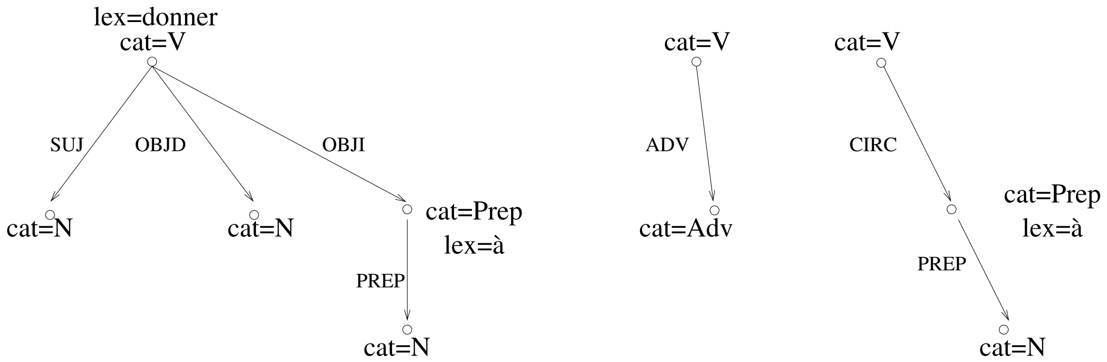


FIG. 4.5: Décomposition d'une structure « élémentaire » en trois structures plus simples

Remarquons que l'ancre lexicale d'une structure élémentaire ne se trouve plus forcément au niveau de la racine de cette dernière, comme, par exemple, dans le cas de l'adverbe ou de la préposition. Nous représenterons l'ancre lexicale par un *nœud noir*, comme dans la figure 4.6.

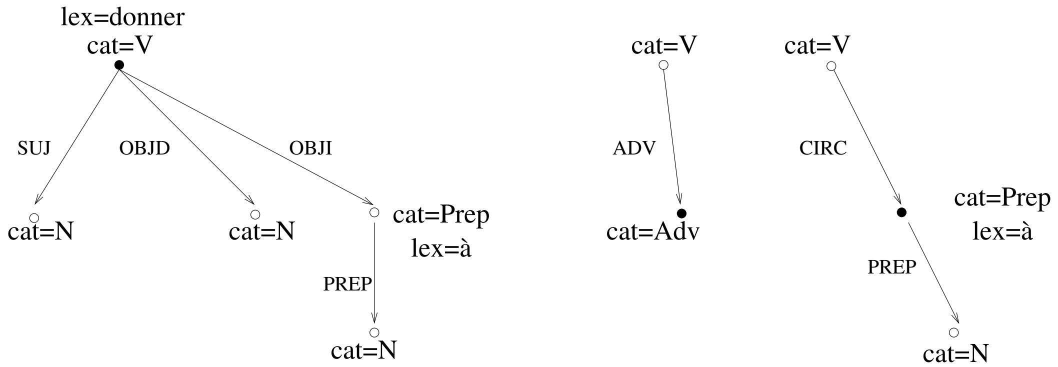


FIG. 4.6: Identification des ancres lexicales dans les structures élémentaires

Pour reprendre la caractérisation en termes de valence active et valence passive que nous avions effectuée dans la section 4.1.2, on pourra dire que la valence passive des modifieurs ou circonstants sera représentée dans les arbres élémentaires associés à ces derniers. Cette différenciation des structures élémentaires selon la nature de leur ancre lexicale permet d'extraire les modifieurs et circonstants du domaine de localité d'un arbre élémentaire. Le phénomène de duplication des dépendances, observé en 4.1.2 lorsque valences actives et passives sont représentées systématiquement, est ainsi évité.

Cette distinction entre arguments et modifieurs se retrouve, en grammaire syntagmatique, dans les grammaires d'arbres adjoints (TAG) où deux types de structures élémentaires de la grammaire sont distinguées : les *arbres initiaux*, qui représentent les structures argumentales complètes d'un prédicat, et les *arbres auxiliaires*, qui sont utilisés pour la représentation de modifieurs, verbes modaux, auxiliaires et verbes à complétives. La figure 4.7 montre dans sa partie gauche des arbres élémentaires TAG et dans sa partie droite nos propres arbres élémentaires. Nous suivrons la terminologie des grammaires d'arbres adjoints et parlerons d'arbres auxiliaires et d'arbres initiaux. Dans le cas des TAG, les arbres initiaux se caractérisent par la présence du prédicat sous forme d'une feuille lexicale et les arbres auxiliaires par la présence d'une feuille, appelée nœud pied (identifié dans le schéma par un astérisque), de même catégorie que la racine de l'arbre. Dans notre cas, les arbres initiaux se caractériseront par la présence de l'ancre (du nœud noir) en position de racine. C'est cette caractéristique qui permet de distinguer les arbres auxiliaires des arbres initiaux.

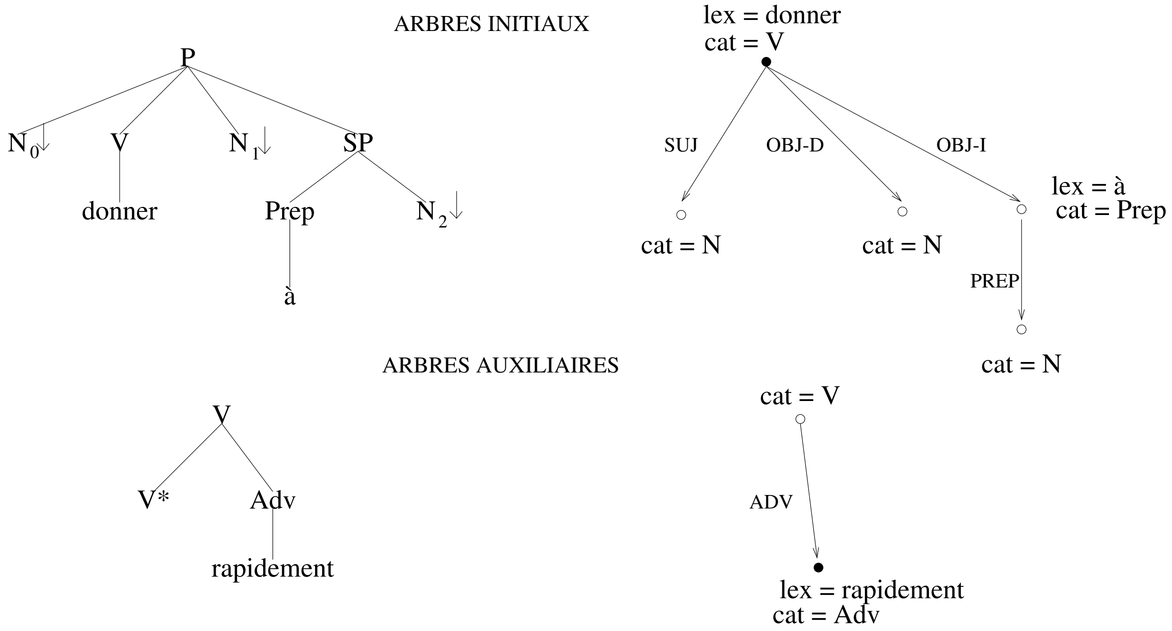


FIG. 4.7: Arbres initiaux et arbres auxiliaires en TAG et dans notre formalisme

4.1.5 Représentation de contraintes d'ordre dans les *AES*

Avant d'aborder le problème complexe de la représentation de contraintes sur l'ordre grammatical dans les *AES*, nous allons nous intéresser au problème plus général de la mise en correspondance d'un arbre de dépendances et d'une chaîne linéaire, autrement dit, à la représentation d'un ordre linéaire sur l'esemble des nœuds d'un arbre de dépendances. A l'issue de cet examen, nous proposerons un moyen de représentation de l'ordre linéaire qui nous servira ensuite à exprimer, dans les *AES*, des contraintes sur l'ordre grammatical.

Représentation de l'ordre linéaire dans les arbres de dépendances

Contrairement au paradigme syntagmatique où l'ordre linéaire et la représentation structurelle d'une phrase sont indissociables, les grammaires de dépendances permettent une représentation structurelle indépendante de l'ordre linéaire, telle que dans les arbres syntaxiques profonds et les arbres syntaxiques de surface de la *TST*. Nous avons représenté côte-à-côte, dans la figure 4.8, un arbre de dépendances et un arbre syntagmatique du groupe nominal *la pompe électrique de le (du) circuit secondaire*. Dans le second, l'ordre entre les mots de la phrase est déterminé mais pas dans le premier, où les dépendants du nœud *pompe*, par exemple, ne sont pas ordonnés : on ne connaît pas la position de l'article *la*, dans la chaîne linéaire, par rapport à l'adjectif *électrique* et à la préposition *de*.

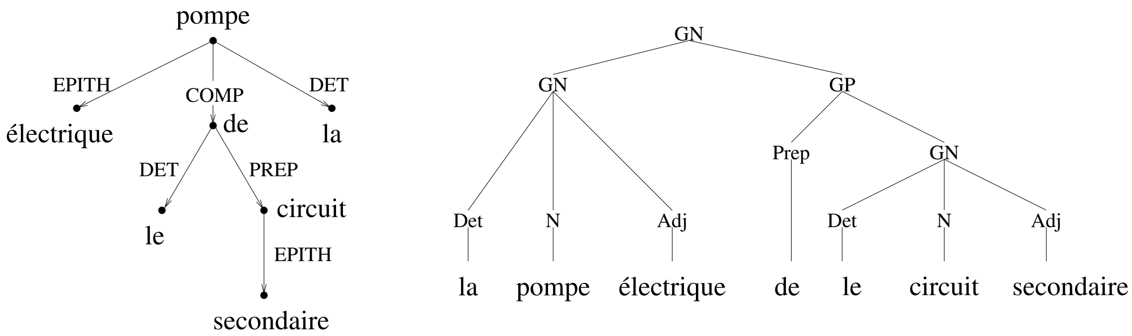


FIG. 4.8: Un arbre de dépendances et un arbre syntagmatique d'un même groupe nominal

Certains enrichissements du formalisme des grammaires de dépendances visant à y inclure des informations sur l'ordre linéaire ont été proposés. Le système de H. Gaifman présenté ci-dessus en offre un exemple : l'ordre linéaire est représenté implicitement par une représentation linéaire des règles de grammaire. De par sa représentation linéaire, la règle $V(N_1, *, Adv, N_2, Prep_1, Prep_2)$ impose un ordre total entre les différentes catégories grammaticales qu'elle manipule : $N_1 < V < Adv < N_2 < Prep_1 < Prep_2$. Une telle représentation mêlant structure linéaire et structure hiérarchique peut être étendue à un arbre de dépendances de profondeur quelconque, comme dans l'exemple suivant[2] :

[2]Pour obtenir une représentation linéaire plus lisible de cette structure, il suffit de remplacer les astérisques par les mots qui leur correspondent :

(la pompe électrique (du (circuit secondaire)))

*pompe (la * électrique du (* circuit (* secondaire)))*

L'ordre des nœuds d'un arbre peut aussi être représenté graphiquement, en ordonnant horizontalement les fils d'un nœud selon leur ordre dans la chaîne linéaire, à l'image de l'arbre de la figure 4.9, où le nœud *électrique* est situé à droite de son gouvernant *pompe* et à gauche de son frère *de*. Lorsque cette convention graphique est respectée, la projection horizontale de l'arbre permet de retrouver l'ordre des mots de la phrase[3].

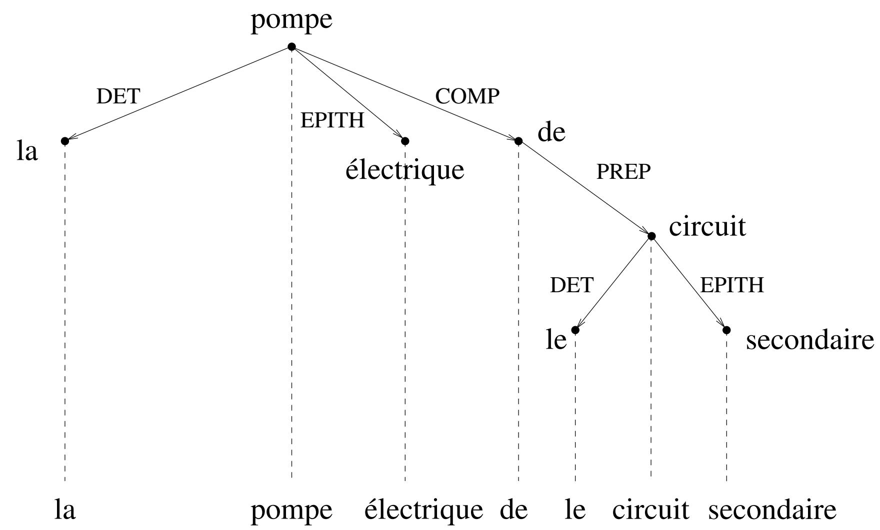


Fig. 4.9: Représentation graphique de l'ordre dans un arbre de dépendance

Dans les deux modes de représentation précédents, l'ordre entre les nœuds d'un arbre revient à indiquer, pour chaque nœud, sa position vis-à-vis de son gouverneur et de ses frères, information que l'on peut aussi représenter symboliquement sous forme de traits. C'est l'approche choisie par [Hellwig, 1986a], qui représente cette information à l'aide d'un trait (*positional feature*) associé à chaque nœud de l'arbre. L'information représentée par ce trait revient en fait à numéroter les fils d'un nœud. Le fils numéro 1 est situé juste après son gouverneur, le fils numéro 2 à la droite du fils numéro 1 ... Le fils numéro -1 est situé juste avant son gouverneur ...

Système de coordonnées linéaires

Nous nous sommes inspiré du système de P. Hellwig pour définir un nouveau système de représentation de l'ordre linéaire dans un arbre de dépendance que nous avons appelé *système de coordonnées linéaires*. Ce système de coordonnées associe à tout nœud d'un arbre les deux

Cette représentation permet de retrouver, grâce au parenthésage, une décomposition possible de la phrase en constituants.

[3]La projection d'un nœud de l'arbre est matérialisée dans la figure 4.9 par un trait vertical en pointillé.

traits *côté* et *sép*. Le trait *côté* indique si un nœud X est situé à droite ou à gauche de son gouverneur et le trait *sép* indique la liste des frères de X (identifiés par leur fonction) situés entre X et son gouverneur. L'exemple précédent a été repris dans la figure 4.10, où chaque nœud a été enrichi de ses coordonnées linéaires. L'information, qui était représentée grâce à une convention graphique dans la figure 4.9, est maintenant représentée à l'aide des coordonnées linéaires. Le nœud *de*, par exemple, est situé à droite de son gouverneur (*côté=d*) et il est séparé de ce dernier par son frère de fonction épithète (*sép*=(EPITH)). Le nœud *électrique* est, lui, placé à la gauche de son gouvernant *pompe* et ne peut être séparé de ce dernier par un autre dépendant de *pompe*.

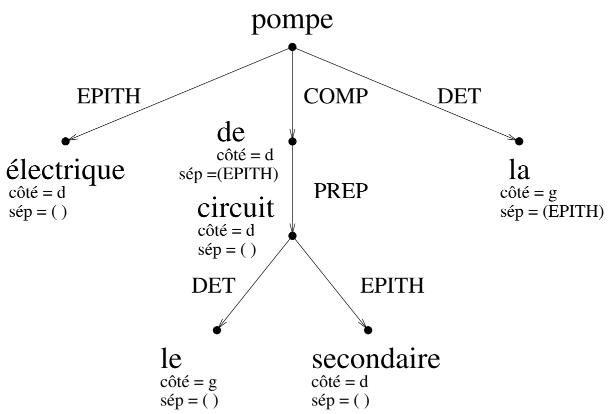


FIG. 4.10: Représentation de l'ordre linéaire dans un arbre de dépendance à l'aide de coordonnées linéaires

Ce système de coordonnées utilise explicitement la notion de fonction syntaxique, matérialisée par les étiquettes des dépendances. Nous avons préféré ce système de représentation à celui de P. Hellwig, qui représente la position des nœuds à l'aide d'indices numériques, car il permet de mieux expliciter le lien entre l'ordre des mots dans la phrase et les fonctions syntaxiques de ces derniers. Il est important de remarquer que les limites du système de représentation de l'ordre que nous proposons dépendent de la richesse de l'ensemble des fonctions syntaxiques. Plus le nombre de fonctions syntaxiques définies sera élevé, plus le sytème de représentation de l'ordre sera fin. Dans le cas de la dépendance épithète, par exemple, si on suppose l'existence d'une seule fonction EPITH, il ne sera pas possible d'imposer un ordre à deux épithètes d'un même substantif se trouvant du même côté de ce dernier. Il ne sera pas possible, en particulier, d'autoriser la séquence *pompe mécanique autorégulatrice* tout en interdisant la séquence *pompe autorégulatrice mécanique*. La prise en compte de cette distinction nécessite un ensemble de fonctions syntaxiques plus riche.

Les trois modes de représentation de l'ordre linéaire que nous venons de passer en revue (le mode linéaire de H. Gaifman, les conventions graphiques et le système de coordonnées linéaires) partagent les deux inconvénients que nous avions mentionnés en 4.1.1 : ils imposent un ordre total aux mots de la phrase et ne permettent pas de représenter des structures non projectives (décrites ci-dessous). Nous allons proposer dans les deux sections suivantes des enrichissements du système de coordonnées linéaires afin de remédier à ces inconvénients.

Représentation d'un ordre linéaire partiel

Le système de coordonnées linéaires offre un moyen simple de représenter un ordre partiel parmi les nœuds d'un arbre. Il suffit pour cela de « sous-spécifier » les traits *côté* et *sép*. Lorsqu'un nœud peut se trouver indiféremment à gauche ou à droite de son gouverneur, on ne spécifiera pas son trait *côté*. Le trait *sép* d'un nœud X, qui indiquait précédemment tous les frères de X séparant ce dernier de son gouverneur, indiquera maintenant uniquement les frères de X devant obligatoirement séparer X de son gouverneur. Les frères de X pouvant être situés indifféremment à gauche ou à droite de X ne seront pas représentés dans le trait *sép* de X. La figure 4.11 présente un arbre dont l'ordre est sous-spécifié, ainsi que les différentes chaînes compatibles avec cet ordre partiel. Dans l'exemple, les nœuds occupant les fonctions ADV et OBJD ne sont pas ordonnés entre eux. Ces contraintes ne définissent donc pas un ordre total, mais un ordre partiel, vérifié par les deux chaînes linéaires représentées dans la figure.

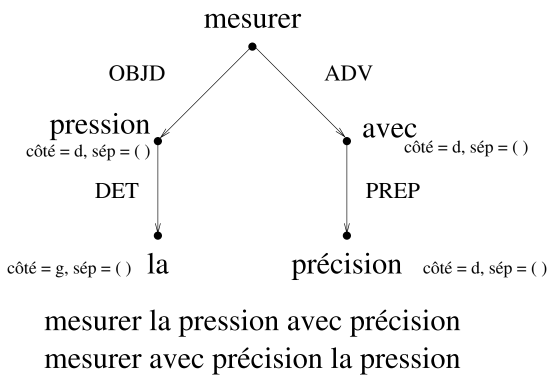


FIG. 4.11: Représentation d'un ordre linéaire partiel

La projectivité

La projectivité est une propriété importante que vérifient certains arbres de dépendances, découverte par Y. Lecerf ([Lecerf, 1960]) . La propriété de projectivité est définie au niveau d'une dépendance, elle s'enonce de la façon suivante :

Une dépendance liant un nœud D à son gouverneur G est projective si les nœuds séparant D de G sont tous des descendants de G.

On dira qu'un arbre est projectif si toutes les dépendances entrant dans sa composition sont projectives.

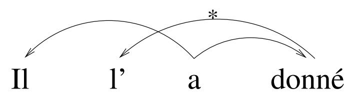
On trouvera, figure 4.12, un exemple de construction non projective[4]. Dans cet exemple, le nœud *l'* est séparé de son gouverneur *donné*, dans la chaîne linéaire, par le nœud *a* qui n'est pas un descendant de *donné* mais son gouverneur[5]. L'importance de la projectivité tient dans le fait qu'elle réduit de façon drastique les façons possibles d'ordonner les nœuds d'un arbre de dépendances[6] et permet de représenter par des moyens simples l'ordre au sein d'un arbre de dépendances. C'est en vertu de ce principe que la représentation de l'ordre linéaire dans un arbre projectif peut se réduire à indiquer, pour chaque nœud, sa position vis-à-vis de son gouverneur et de ses frères ; ces deux informations sont par contre insuffisantes pour représenter l'ordre linéaire dans un arbre non projectif. Dans l'exemple de la figure 4.12, le fait de savoir que le nœud *l'* se trouve à gauche de son gouverneur n'est plus suffisant pour le placer correctement dans la chaîne linéaire, car il est maintenant nécessaire de le situer vis-à-vis d'autres nœuds, notamment des nœuds *a* et *il*. La représentation de l'ordre linéaire dans un arbre non projectif nécessite par conséquent un système plus riche que le système de coordonnées linéaires proposé ci-dessus.

La projectivité est une contrainte intéressante pour le traitement automatique, du fait qu'elle réduit l'ensemble des arbres syntaxiques pouvant correspondre à une phrase. Mais c'est malheureusement un principe trop restrictif pour représenter certaines constructions linguistiques telles que celles données dans l'exemple de la figure 4.12 ou certaines subordonnées relatives telles que *l'histoire dont je connais la fin*[7] où le pronom relatif *dont* est séparé de son gouverneur *fin* par le gouverneur de ce dernier.

---

[4]La dépendance non projective est marquée d'un astérisque. On parlera indifféremment de dépendance non projective ou de *nœud non projectif*, ce dernier étant le nœud situé à l'extrémité de la dépendance non projective.

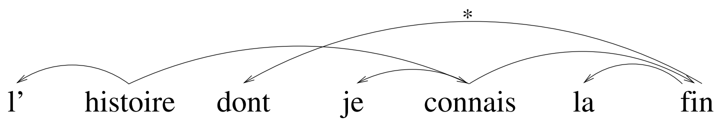
[5]Il existe un moyen graphique simple de déceler les structures non projectives en représentant dans la chaîne linéaire les dépendances entre mots au-dessus de ces derniers. Dans les cas non projectifs, deux phénomènes se produisent : le croisement de dépendances et/ou le passage d'une dépendance au-dessus du nœud racine.



[6][Béringer, 1988] a effectué une comparaison du nombre d'arbres de N nœuds pouvant être associés à une phrase de N mots selon que le principe de projectivité est pris en compte ou pas. Les résultats sont repris dans le tableau suivant.

nb nœuds	3	4	5	6	7	8	9	10
arbres n-proj	9	64	625	7776	117649	2.10^6	43.10^6	1.10^9
arbres proj	7	30	143	728	3876	21318	1200001	690690
rapport	1	2	4	11	30	98	358	1448

[7]



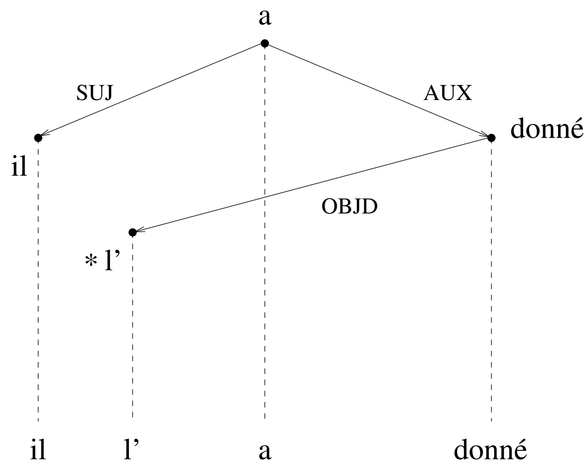


Fig. 4.12: Un arbre non projectif

La pseudo-projectivité

Bien que la présence de constructions non projectives soit attestée en linguistique, ces dernières sont en nombre restreint et ne représentent qu'un sous-ensemble des structures non projectives possibles. L'idée est alors d'essayer de caractériser les structures non projectives les plus communes apparaissant dans la langue, afin d'enrichir notre formalisme et de représenter ces dernières sans toutefois chercher à représenter tous les cas de non projectivité. C'est dans cet esprit que nous proposons la notion de *pseudo-projectivité*, qui permet de caractériser certains cas de non projectivité. Nous avons élaboré la notion de pseudo-projectivité par l'observation des exemples de structures linguistiques non projectives apparaissant dans [Mel'čuk, 1988a] et [Marcus, 1965], qui vérifient toutes la propriété de pseudo-projectivité. Cette dernière nous apparaît comme une condition raisonnable à imposer aux structures linguistiques, sans, bien entendu, faire l'hypothèse de son universalité.

La définition que nous allons donner de la pseudo-projectivité se fonde sur la notion de *gouverneur linéaire* d'un nœud et sur l'opération de *relevage* d'une dépendance, définies de la façon suivante :

Le gouverneur linéaire d'un nœud X est l'ancêtre de X de degré le plus élevé séparant X de son gouverneur. Dans le cas d'une dépendance projective, le dépendant ne peut être séparé de son gouverneur par des ancêtres de ce dernier. On dira dans ce cas que le gouverneur linéaire et le gouverneur hierarchique se confondent.

Relever une dépendance P, liant un nœud D à son gouverneur, sur un nœud X consiste à éliminer P et à établir une nouvelle dépendance entre X et D, orientée de X vers D.

La pseudo-projectivité s'énonce de la façon suivante[8] :

[8]Cette définition de la pseudo-projectivité nous a été proposée par Sylvain Kahane

Un arbre est pseudo-projectif si, en relevant chaque dépendance, liant un nœud D à son gouverneur, sur le gouverneur linéaire de D, on obtient un arbre projectif.

On remarquera que les opérations de relevage dont il est question dans la définition de la pseudo-projectivité, ne s'appliquent qu'aux dépendances non projectives. En effet, dans le cas d'une dépendance projective, les notions de gouverneur linéaire et de gouverneur hierarchique se confondent, le relevage de la dépendance n'est donc pas pertinent. Dans le cas d'un arbre projectif, aucune opération de relevage ne doit être effectuée pour obtenir un arbre projectif ! un arbre projectif est donc pseudo-projectif.

Nous avons représenté dans la figure 4.13 un exemple de construction violant la contrainte de pseudo-projectivité. Dans cet exemple, le nœud c est séparé de son gouverneur d par le nœud a qui n'est pas un descendant de d. Le nœud c n'est donc pas pseudo-projectif. De plus, il n'existe pas d'ancêtres de d situés entre d et c, la dépendance liant c à d ne peut donc être relevée. Il n'est par conséquent pas possible, par relevage des dépendances, d'obtenir un arbre projectif, l'arbre n'est donc pas pseudo-projectif.

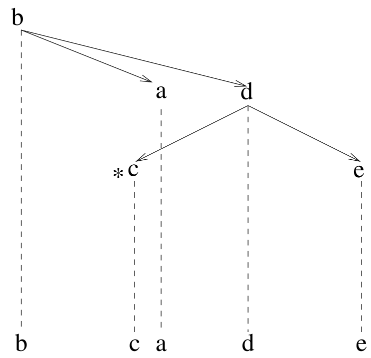


Fig. 4.13: Une structure violant la contrainte de pseudo-projectivité

Nous avons illustré différents cas de pseudo-projectivité dans la figure 4.14 où est représenté un arbre de dépendance comprenant un nœud non projectif (marqué d'un astérisque) ainsi que les cinq différentes positions que le mot correspondant à ce nœud peut occuper dans la chaîne linéaire sans violer l'hypothèse de pseudo-projectivité. C'est en vertu de la pseudo-projectivité que la projection de i ne peut être située entre e et g ni entre a et c. L'ordre des quatre dernières séquences qui correspondent à des structures non projectives ne peut être représenté à l'aide du système de coordonnées linéaires présenté ci-dessus.

La notion de gouverneur linéaire d'un nœud, définie ci-dessus, offre un moyen naturel de classer différents cas de pseudo-projectivité. En fonction du niveau hiérarchique du gouverneur linéaire d'un nœud donné, on associera à ce dernier un certain *niveau de pseudo-projectivité*. Dans le

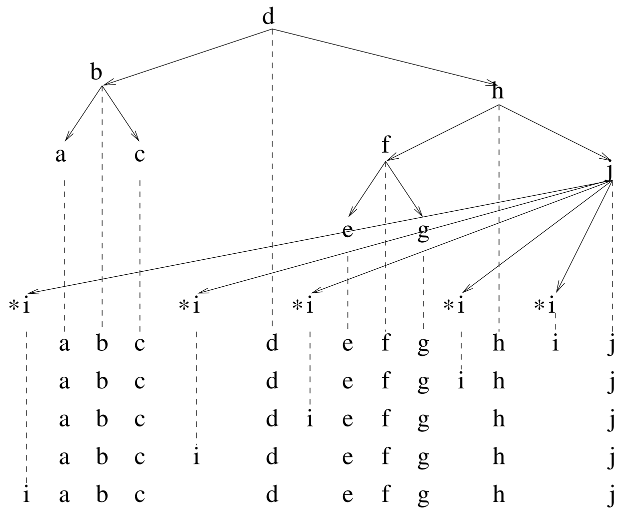


FIG. 4.14: Diverses configurations pseudo-projectives

cas d'un nœud projectif, lorsque le gouverneur linéaire d'un nœud X n'est autre que son gouverneur hiérarchique, le niveau de pseudo-projectivité associé à X sera égal à zero. Lorsque le gouverneur linéaire de X est le gouverneur de son gouverneur, le niveau de pseudo-projectivité de X est égal à l'unité et ainsi de suite. La projectivité est ainsi ramenée à un cas particulier de pseudo-projectivité, la pseudo-projectivité de niveau zero.

Nous avons repris dans la figure 4.15 les exemples de la figure 4.14, en indiquant, pour chacune des chaînes linéaires, le niveau de pseudo-projectivité associé à la dépendance non projective. Dans la dernière chaîne linéaire de la figure 4.15 (i a b c d e f g h j), le gouverneur linéaire du nœud non projectif i est son ancêtre de niveau trois (d), c'est pourquoi le niveau de non projectivité associé à i est égal à deux.

En vertu de la définition de la pseudo-projectivité, il est possible d'engendrer, par relevage des dépendances, à partir d'un arbre pseudo-projectif, un arbre projectif. Nous somme donc en présence de deux arbres de dépendances, une *représentation hiérarchique de l'arbre* (l'arbre de dépendances « normal ») et une *représentation projective de l'arbre*, dans laquelle un nœud ne se trouve plus sous la dépendance de son gouverneur hiérarchique mais sous celle de son gouverneur linéaire (voir figure 4.16)[9]. Dans le cadre de la représentation projective, on parlera de gouverneurs linéaires, de dépendants linéaires et de frères linéaires.

[9]Si la représentation hierarchique se confond avec la *SSyntS* de la *TST*, la structure projective ne possède par contre pas d'équivalent dans la *TST* et apparaît comme une une *SSyntS* mal formée. Ainsi, la dépendance étiquetée OBJD liant l'auxiliaire à l'objet direct dans la figure 4.15 est incorrecte vis-à-vis de la *TST*. Nous aurions dû, en toute rigueur, définir de nouvelles étiquettes de dépendance pour les représentations projectives des *SSyntS*.

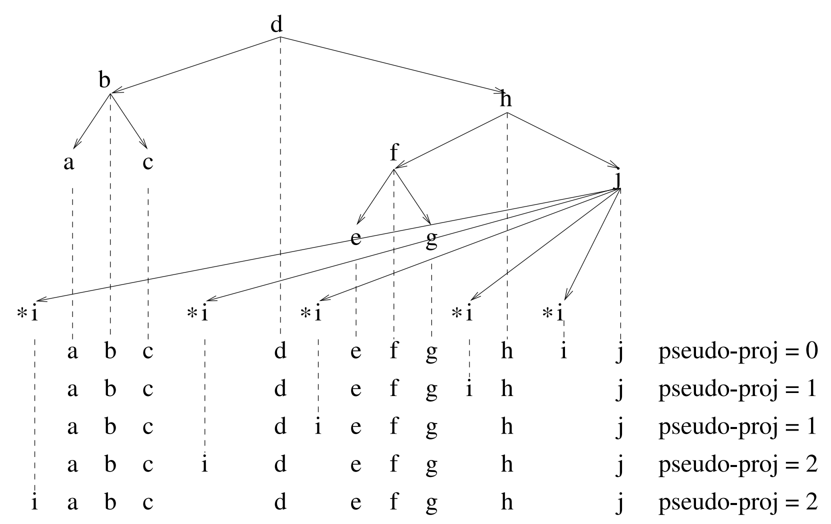


Fig. 4.15: Niveaux de pseudo-projectivité

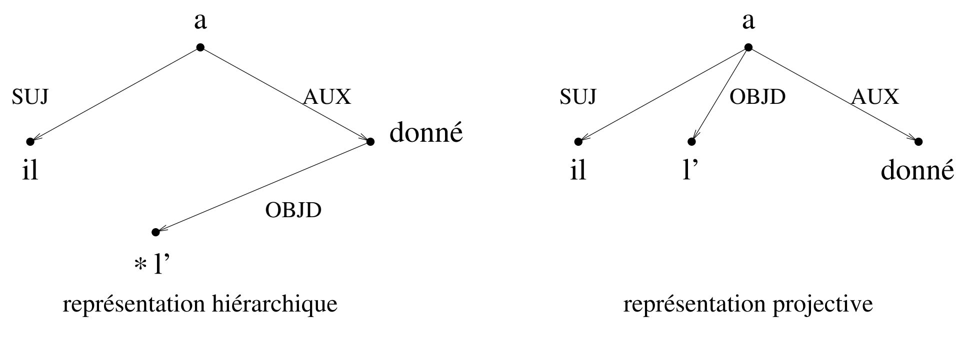


Fig. 4.16: Représentations hierarchique et projective d'une structure de dépendance

Système de coordonnées pseudo-projectives

La notion de gouverneur linéaire (ou la notion équivalente de niveau de pseudo-projectivité) a été introduite dans le but de proposer un nouveau système de coordonnées linéaires. Ce dernier, que l'on appellera *système de coordonnées pseudo-projectives*, permet d'associer à un arbre pseudo-projectif une chaîne linéaire unique, ce qui ne pouvait être fait avec le système de coordonnées linéaires défini ci-dessus. On indiquera pour chaque nœud de l'arbre :

- son niveau de pseudo-projectivité (le trait *ps-p*),
- sa position (gauche ou droite) par rapport à son gouverneur linéaire (le trait *côté*),
- l'ensemble de ses frères linéaires situés entre son gouverneur linéaire et lui (le trait *sép*).

Dans l'exemple de la figure 4.16, les coordonnées pseudo-projectives du nœud correspondant au clitique *l'* (*ps-p* = *2*, côté = g et *sép* = ()), indiquent que le gouverneur linéaire du nœud est son ancêtre de niveau deux (le verbe avoir), que le pronom se trouve à la gauche de son gouverneur linéaire, et qu'il ne peut être séparé de ce dernier par d'autres fils linéaires du verbe avoir.

La représentation d'un ordre partiel dans un arbre pseudo-projectif peut être obtenue, à l'instar d'un ordre partiel dans un arbre projectif, en sous-spécifiant les traits *côté* et *sép*.

Représentation de contraintes d'ordre dans les *AES*

Le système de coordonnées pseudo-projectives, permettant la représentation d'un ordre partiel dans un arbre de dépendances pseudo-projectif, va permettre de spécifier des *contraintes d'ordre linéaire* dans les *AES*. A chaque nœud des *AES* seront associées des coordonnées pseudo-projectives, permettant de contraindre sa position vis-à-vis des autres nœuds de l'arbre élémentaire. Nous avons représenté dans la figure 4.17 un *AES* dont les nœuds sont enrichis de coordonnées pseudo-projectives. Les contraintes apparaissant au niveau de l'objet direct, par exemple, indiquent que ce dernier doit être situé à droite de son gouverneur, et qu'il peut être séparé de ce dernier par un objet indirect, des adverbes ou des compléments circonstanciels. Il est important de noter que les contraintes d'ordre linéaire apparaissant dans un *AES* ne font pas toujours référence à des nœuds de l'*AES*. Dans la figure 4.17, par exemple, la position de la préposition *à* est définie par rapport à certains nœuds de l'*AES* (l'objet direct et le verbe) mais aussi par rapport à des nœuds non représentés dans l'arbre élémentaire (de potentiels adverbes et compléments circonstanciels). Les contraintes d'ordre linéaire associées à un nœud d'un *AES* permettent donc de le positionner dans un contexte plus large que l'*AES*. La prise en compte de ces contraintes se fera lorsque l'*AES* participera à une opération d'attachement, comme nous le verrons en 4.2.

On peut remarquer que les contraintes d'ordre associées aux nœuds de l'arbre de la figure 4.17 ne sont pas particulières à ce dernier. Le fait que le sujet soit situé à gauche du verbe et qu'il puisse être séparé de ce dernier par un adverbe de négation est une information partagée par la majorité des *AES* associés à un verbe. C'est pourquoi les contraintes d'ordre ne sont pas définies au

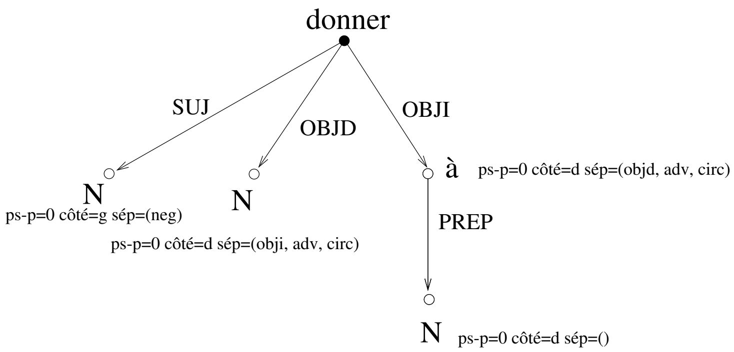


FIG. 4.17: Un *AES* enrichi de coordonnées pseudo-projectives

niveau des *AES* mais au niveau des dépendances, permettant ainsi une meilleure factorisation des informations sur l'ordre linéaire. A chaque type de dépendance, seront associées des *contraintes positionnelles* consistant à spécifier la valeur des trois traits *ps-p*, *côté* et *sép* du nœud situé à l'extrémité de la dépendance. Les contraintes d'ordre peuvent néanmoins être redéfinies au niveau des arbres élémentaires pour prendre en compte d'éventuelles idiosyncrasies lexico-syntaxiques. Les informations représentées par les deux traits *côté* et *sép* peuvent être directement mises en relation avec les règles syntagme et les règles d'ordonnancement de la *TST* présentées en 3.2. La valeur du trait *côté* est représentée, dans la *TST*, dans les règles syntagme et la valeur du trait *sép* par les règles d'ordonnancement. Les contraintes d'ordre linéaire représentées au niveau des *AES* sont en fait l'accumulation des contraintes positionnelles associées aux différentes dépendances entrant dans la composition de l'*AES*.

4.1.6 Principes de bonne formation des *AES*

La simplicité des *AES* présentés jusque-là est assez trompeuse et l'écriture d'*AES* plus complexes se heurte à un certain nombre de difficultés. Le problème majeur concerne l'étendue des arbres élémentaires : quels nœuds et quelles dépendances doivent apparaître dans un *AES* particulier ? Afin de guider la construction d'*AES*, nous allons proposer des *principes de bonne formation* des *AES*. Les principes que nous allons proposer ne permettent pas toujours de décider de l'étendue de certains *AES*, comme nous le verrons en annexe B. Ils nous a néanmoins semblé important d'essayer de les formaliser, quitte à les modifier par la suite.

Les deux premiers principes de bonne formation ont déjà été énoncés : le premier impose aux *AES* d'avoir une structure d'arbre et le second impose qu'à un *AES* soit associé une ancre lexicale unique. Nous avons représenté dans la figure 4.18 un *AES* associé au verbe *donner*. L'écriture d'un tel arbre ne pose pas de problèmes majeurs, la définition informelle que nous avons donnée des *AES* s'avère suffisante : le nœud correspondant au verbe (l'ancre lexicale) est représenté à la racine de l'arbre; ce nœud a pour fils le sujet, l'objet direct et la préposition introduisant l'objet indirect. Nous avons de plus représenté graphiquement les contraintes morphologiques, sémantiques et lexicales existant entre les nœuds sous la forme de dépendances morphologiques sémantiques et lexicales. Dans l'exemple la dépendance morphologique représente le phénomène d'accord entre le verbe et le sujet, les dépendances sémantiques représentent les contraintes sémantiques qu'impose le verbe à son sujet, son objet direct et son objet indirect. L'unique dépendance lexicale concerne la nature lexicale de la préposition (*à*) introduisant l'objet indirect.

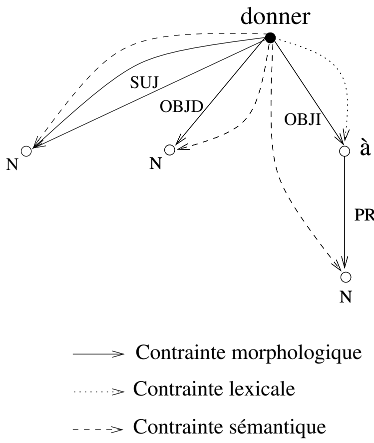


FIG. 4.18: Différentes dépendances liant les nœuds d'un *AES*

On pourra remarquer que les différentes dépendances non syntaxiques représentés dans la figure 4.18 ont pour origine l'ancre lexicale de l'*AES*. C'est là le troisième principe de bonne formation des *AES* :il impose que toutes les dépendances non syntaxiques aient l'ancre de l'*AES* pour origine ou pour extrémité. Ce principe s'explique intuitivement par le fait qu'un *AES* représente un contexte d'occurrence d'un lexème particulier : l'ancre de l'*AES*. Les contraintes représentées dans l'*AES* doivent par conséquent être liées à ce lexème. Il n'y a pas de raison de représenter, par exemple, dans un *AES* correspondant à un verbe, la contrainte d'accord entre le sujet du verbe et son déterminant. Une telle contrainte est indépendante du verbe et rien ne justifie sa présence dans l'*AES* du verbe. Cette condition ne concerne que les dépendances non syntaxiques. On pourra remarquer en effet que la dépendance PREP, liant dans l'*AES* de la figure 4.18 la préposition *à* à l'objet indirect, est représentée dans l'*AES* bien qu'elle ne soit pas directement liée à l'ancre *donner* de l'*AES*. Son introduction dans l'arbre est nécessaire pour satisfaire le premier principe de bonne formation garantissant la structure arborescente de l'*AES*. Si nous n'avions pas introduit la dépendance entre la préposition et l'objet indirect, tout en représentant l'objet indirect au sein de l'*AES*, ce dernier aurait perdu sa structure d'arbre.

Nous avons représenté dans la figure 4.19 un *AES* ayant un participe passé pour ancre lexicale. Afin de représenter la contrainte sémantique qu'impose le participe passé au sujet, le nœud correspondant à ce dernier, bien qu'il ne soit pas directement lié à l'ancre de l'*AES* par une dépendance syntaxique, est représenté dans l'*AES*, en vertu du troisième principe. L'introduction de ce nœud nécessite la représentation, dans l'*AES*, de la dépendance *sujet* reliant l'auxiliaire et le sujet, sans quoi l'*AES* aurait perdu sa structure d'arbre, violant le premier principe de bonne formation. On pourra remarquer que la contrainte d'accord entre le sujet et l'auxiliaire n'a pas été représentée dans l'*AES* du participe passé, en accord avec le troisième principe de bonne formation.

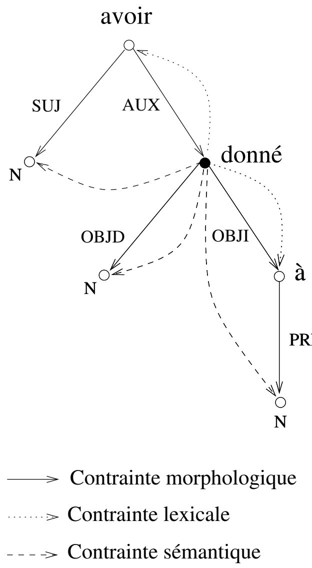


FIG. 4.19: Un *AES* complexe

Une conséquence importante de ces principes de bonne formation est qu'une dépendance syntaxique peut être représentée dans plusieurs *AES* différents. Ainsi, la dépendance liant un auxiliaire à un participe passé sera représentée aussi bien dans l'*AES* associé au participe passé, comme nous l'avons fait dans la figure 4.19, que dans l'*AES* associé à l'auxiliaire. Ces trois principes de bonne formation ne permettent pas toujours de définir naturellement la structure des arbres élémentaires. Nous verrons en B des cas d'*AES* problématiques qui se posent lors de l'écriture d'une grammaire.

4.1.7 Représentation des *AES* à l'aide de structures de traits

La représentation de certaines connaissances sous forme de traits dans les *AES* a été amorcée dès le début de ce chapitre en enrichissant les nœuds des *AES* d'ensembles de couples attribut, valeur. Nous allons généraliser ici l'emploi de structures de traits pour en faire le moyen unique de représentation des structures lexico-syntaxiques, à l'image de plusieurs formalismes grammaticaux récents, tels que FUG ([Kay, 1984]), HPSG ([Pollard & Sag, 1994]) et LFG ([Bresnan, 1982]).

Nous n'introduirons pas en détail ici les structures de traits comme outil de représentation de connaissances linguistiques, ni les nombreuses extensions que cet outil a connu (traits complexes, structures réentrantes, disjonction de valeurs ...). On trouvera des descriptions de ce mode de représentation dans [Abeillé, 1993, Sabah, 1988] ou [Shieber, 1986]. Les bénéfices attendus du recours à ce mode de représentation sont doubles. Il permet d'une part de représenter toutes les informations composant les *AES* (syntaxiques, lexicales, sémantiques et morphologiques) dans un formalisme unique au lieu des diverses règles de la *TST*. Il permet d'autre part, d'utiliser l'opération d'unification pour la composition d'*AES*, comme nous le verrons dans la section suivante sur l'opération d'attachement.

La structure arborescente des *AES* ainsi que les contraintes d'ordre linéaires sont directement représentables par des structures de traits, comme l'illustre l'exemple de la figure 4.20. Les arcs des *AES* sont représentés sous forme de traits à valeur complexe dont l'attribut est constitué par l'étiquette de l'arc. Le trait *sép* (défini en 4.1.5) permettant de représenter les dépendances pouvant séparer un nœud de son gouverneur sera représenté par un trait de type particulier dont la valeur est constituée d'une liste de symboles (*trait à valeur de liste simple*).

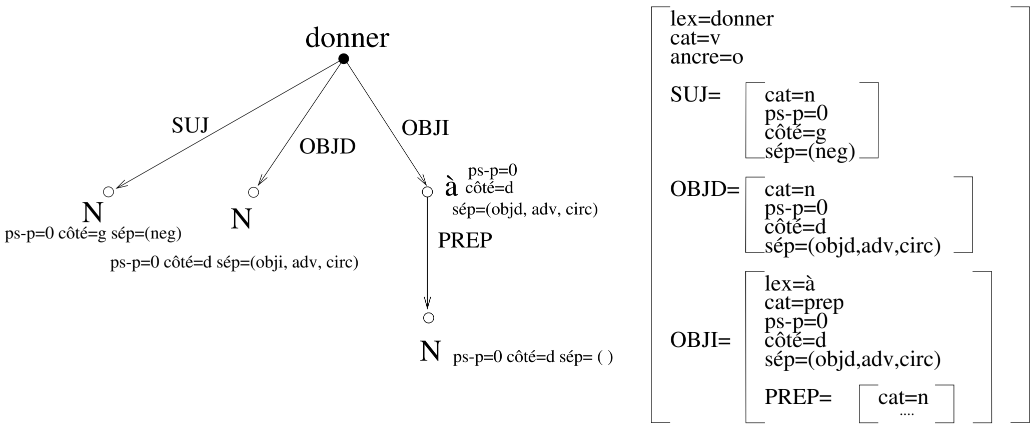


FIG. 4.20: Représentation d'un *AES* sous forme d'une structure de traits

L'existence de dépendances répétables pose un problème de représentation, du fait que deux traits d'une même structure ne peuvent avoir le même nom d'attribut. De tels cas seront pris en compte par des *traits à valeur de liste complexe*, à l'image de la figure 4.21 où l'attribut EPITH a pour valeur deux structures complexes (deux nœuds).

4.2 L'opération d'attachement

L'opération d'attachement permet de combiner deux arbres pour n'en former qu'un. Elle consiste à unifier la racine du premier arbre (A_1) avec un nœud du second (A_2), appelé *site d'attachement*. Nous dirons, à l'issue de l'opération, que A_1 est attaché *dans* A_2. Les arbres combinés par attachement peuvent être des *AES* ou des sous-arbres plus étendus, comme l'illustre la figure 4.22.

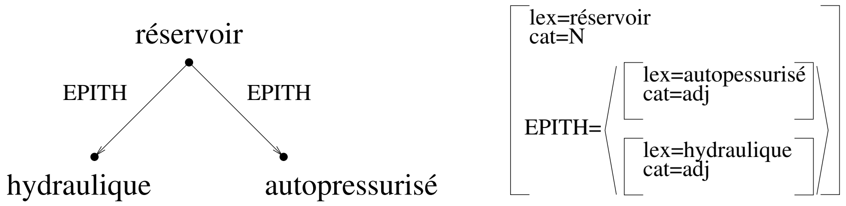


FIG. 4.21: Représentation de dépendances répétables au sein d'une structure de traits

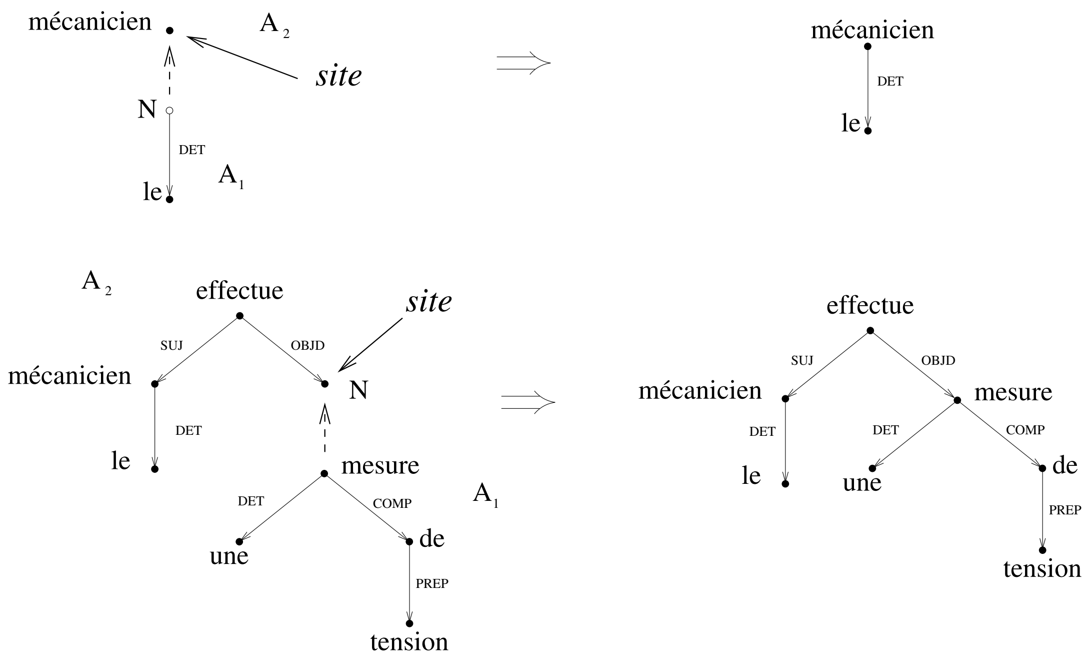


FIG. 4.22: Deux exemples d'attachement

L'opération d'attachement doit être envisagée du point de vue de l'analyse : elle permet de combiner deux arbres correspondant à deux segments contigus de la phrase soumise à l'analyse, pour constituer un arbre correspondant à la concaténation des deux segments de la phrase, à l'image de la figure 4.23, où l'arbre A_1 est attaché dans l'arbre A_2 pour former l'arbre A_3. Dans l'exemple de la figure 4.23, l'arbre A_1 est situé à la droite de l'arbre A_2, conséquence du fait que S_1 est situé à droite de S_2 ; on parlera dans ce cas d'*attachement à droite*. Lorsque l'arbre A_1 est situé à la gauche de A_2, on parlera d'*attachement à gauche*. On peut voir l'opération d'attachement gauche de deux arbres comme l'établissement d'une dépendance gauche[10] entre deux mots des segments de phrase correspondant aux deux arbres, et symétriquement, l'opération d'attachement droit comme l'établissement d'une dépendance droite entre deux mots des deux segments. Dans l'exemple de gauche de la figure 4.22, une dépendance est établie entre le verbe *effectuer* appartenant au sous-arbre gauche et le nom *mesure* appartenant au sous-arbre gauche. Nous appellerons cette dépendance, établie à l'issue de l'opération d'attachement, la *dépendance d'attachement* établie par l'opération d'attachement. Elle est représentée en pointillés dans les figures 4.23 et 4.24.

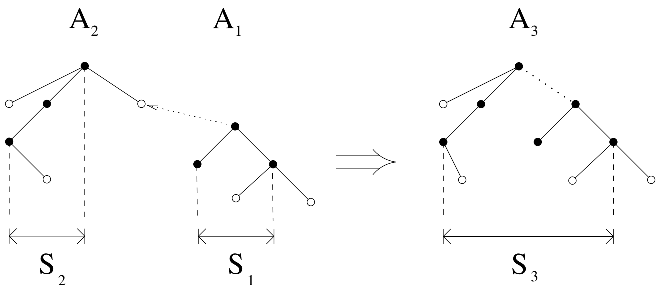


FIG. 4.23: Attachement à droite de deux arbres

Nous avons représenté dans la figure 4.24 l'attachement à droite de deux arbres correspondant à deux segments de la phrase *le mécanicien vérifie la troisième pompe*. Le premier segment est constitué des trois premiers mots de la phrase et le second des trois derniers. A l'issue de l'opération, un arbre correspondant à la phrase complète est constitué. La dépendance d'attachement est une dépendance OBJD, établie entre le verbe *vérifier* et son objet *pompe*.

Domaine noir d'un arbre

Nous avons imposé aux *AES* de posséder une ancre lexicale représentée par un nœud noir. Lors de l'anayse, au fur et à mesure que des *AES* sont combinés par attachement, le nombre de nœuds noirs des arbres créés augmente. Les nœuds noirs d'un arbre correspondent donc aux mots de la phrase déjà traités par le processus d'analyse. Nous appellerons la partie de l'arbre composée exclusivement de nœuds noirs le *domaine noir* de l'arbre. Un arbre créé par attachements successifs

[10] Une dépendance gauche est une dépendance dont le dépendant est situé à gauche du gouverneur.

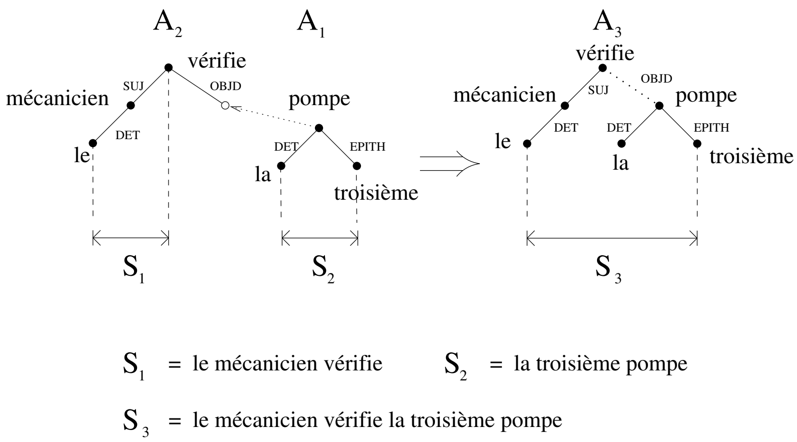


FIG. 4.24: Un exemple d'attachement à droite

comportera aussi un certain nombre de nœuds blancs qui correspondent à un contexte potentiel qui n'a pas encore été reconnu dans la phrase analysée. Nous avons repris dans la figure 4.25 le segment de phrase *le mécanicien vérifie* et l'arbre lui correspondant. On pourra remarquer que seuls les nœuds noirs de la *SSyntS* correspondent à des mots du segment de phrase. Le nœud blanc, correspondant à l'objet direct, ne peut être mis en correspondance avec un mot du segment de phrase ; il a été introduit dans l'arbre à la suite de l'attachement de l'*AES* correspondant au verbe *vérifier* mais n'a pas encore été reconnu dans la phrase.

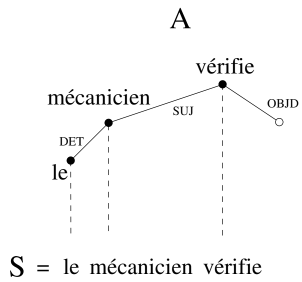


FIG. 4.25: Domaine noir d'un arbre

4.2.1 Détails de l'opération d'attachement

Les principes généraux de l'opération d'attachement et son rôle vis-à-vis de l'analyse ont été décrits ci-dessus. La réalisation de cette opération est toutefois assez complexe et nous allons la décrire ici sous forme de trois conditions devant être vérifiées pour que l'opération d'attachement aboutisse :

1. Unification de la racine de A_1 et d'un site de A_2

Une première condition qui doit être vérifiée lors de l'attachement d'un arbre A_1 dans un arbre A_2 est celle de l'unification de la racine de A_1 et d'un site de A_2. Nous ne décrirons pas ici l'opération d'unification de structures de traits complexes. Cette description pourra être trouvée dans [Shieber, 1986] ou [Gazdar & Mellish, 1989]. L'opération d'unification, grâce à laquelle nous effectuons la combinaison d'arbres, se distingue néanmoins de l'opération d'unification classique dans la prise en compte des traits à valeur de liste, que nous avions introduit en 4.1.7 pour représenter les dépendances répétables. L'unification de traits à valeur de listes complexes consiste à réaliser la concaténation des deux listes de traits, comme l'illustre la figure 4.26[11].

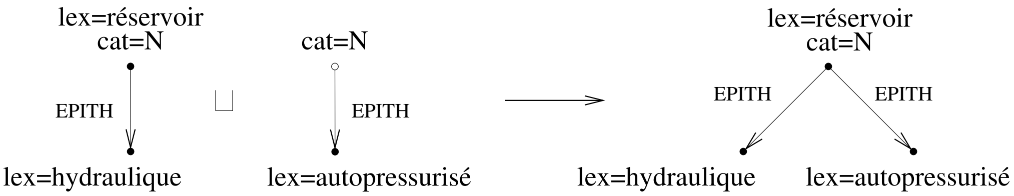


FIG. 4.26: Unification de dépendances répétables

2. Connexité du domaine noir

   A l'image des différents exemples d'attachement proposés ci-dessus, le domaine noir d'un arbre résultant d'une opération d'attachement doit être *connexe*. Lors de l'attachement de deux arbres une dépendance devra lier les domaines noirs des deux arbres. Ainsi, du fait que tous les *AES* possèdent un nœud noir et que l'opération d'attachement garantit la connexité du domaine noir, tout arbre de surface formé par attachements successifs aura un domaine noir connexe. La dépendance liant les domaines noirs des deux arbres participant à l'opération d'attachement est la dépendance d'attachement définie ci-dessus. La figure 4.27 représente un arbre dont le domaine noir n'est pas connexe, correspondant au segment de phrase *le mécanicien vérifie la troisième*. Un tel arbre ne pourra être construit par attachements successifs d'*AES*. L'opération d'attachement impose donc implicitement qu'à une partie de la phrase analysée corresponde un arbre dont le domaine noir est connexe. Cette contrainte va avoir des conséquences importantes sur le processus d'analyse décrit en 5.1.

3. Vérification des contraintes positionnelles associées à la dépendance d'attachement

   Nous avons proposé en 4.1.5, grâce au système de coordonnées pseudo-projectives, une façon de représenter des contraintes sur l'ordre linéaire dans les *AES*. Ces contraintes sont prises en compte durant l'opération d'attachement : lors de l'attachement de deux arbres, les contraintes linéaires attachées au nœud situé à l'extrémité de la dépendance d'attachement devront être satisfaites. Ainsi, lors d'une opération d'attachement de deux segments contigus d'une phrase, les contraintes linéaires d'une seule dépendance sont vérifiées : la dépendance

[11]On trouvera une extension similaire de l'opération d'unification dans [Boyer & Lapalme, 1985].

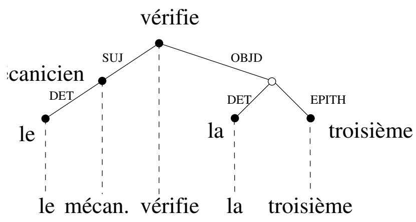


FIG. 4.27: Un arbre présentant un domaine noir non connexe

d'attachement. Dans l'exemple d'attachement de la figure 4.24, cette vérification consiste à s'assurer que le nœud *pompe*, situé à l'extrémité de la dépendance d'attachement, se situe dans la chaîne linéaire à droite du verbe, et qu'il n'est séparé de ce dernier que par des dépendants du verbe apparaissant dans le trait *sép*.

Nous avons représenté dans la figure 4.28 un second exemple d'attachement, correspondant à l'attachement de l'*AES* associé à l'adverbe *méticuleusement* dans l'arbre correspondant au segment de phrase *le mécanicien vérifie la troisième pompe*, pour constituer la phrase *le mécanicien vérifie la troisième pompe méticuleusement*. Dans ce cas, la dépendance d'attachement appartient à l'*AES* de l'adverbe, qui sera attaché dans un second arbre, contrairement au cas de la figure 4.24 où la dépendance d'attachement appartenait à l'*AES* dans lequel à lieu l'attachement. Les contraintes positionnelles associées au nœud *méticuleusement*, qui est situé à l'extrémité de la dépendance d'attachement, impose que ce dernier soit situé à la droite du verbe et qu'il peut être séparé de ce dernier par un objet direct.

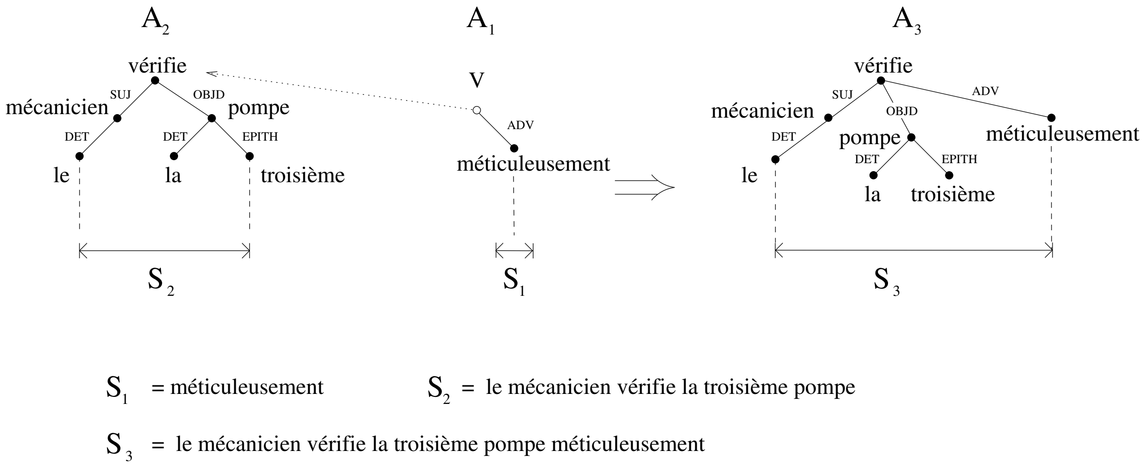


FIG. 4.28: Attachement d'un adverbe

La mise en oeuvre de l'opération d'attachement se décompose en trois étapes : une étape de sélection des sites potentiels d'attachement dans A_2 de façon à garantir la connexité du domaine noir de l'arbre créé, une étape de vérification des conditions sur l'ordre linéaire de la dépendance d'attachement et finalement l'unification de la racine du premier arbre avec un site du second. Dans la mesure où un arbre peut avoir plusieurs sites d'attachement, une opération d'attachement peut aboutir à la création de plusieurs arbres, comme l'illustre la figure 4.29 où l'attachement des arbres correspondant respectivement aux deux segments *le clapet d'isolement* et *de la pompe* donne naissance à deux arbres.

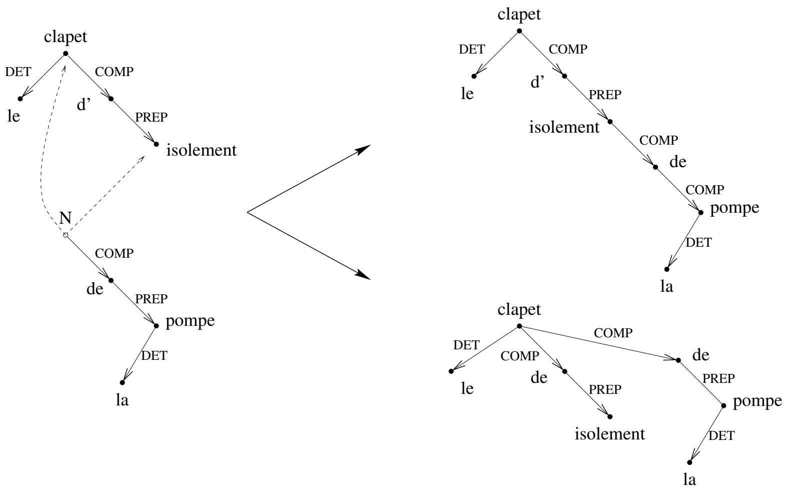


FIG. 4.29: Opération d'attachement aboutissant à la création de deux arbres

4.2.2 Opérations de combinaison d'arbres définies dans d'autres formalismes

Toutes les grammaires arborescentes, que ce soit des grammaires de dépendances ou des grammaires de constituants, définissent des opérations de combinaison d'arbres élémentaires. Dans le cadre des grammaires de dépendances, les opérations de combinaison proposées aussi bien par [Hellwig, 1986a, Fraser, 1989, Sleator & Temperley, 1991] que par [Gaifman, 1965] ont la particularité de combiner les arbres par leurs extrémités : la racine du premier avec une feuille du second. Notre opération d'attachement s'en distingue en autorisant l'attachement de la racine du premier avec un nœud quelconque du second, feuille ou pas, à l'image de l'exemple de la figure 4.29. Cette particularité de notre opération d'attachement est une conséquence du choix que nous avons effectué en 4.1.4 de ne pas représenter dans un *AES* les modifieurs de l'ancre. Du fait de ce choix, il devient nécessaire de pouvoir ajouter à un nœud des dépendants (les modifieurs ou circonstants) même lorsque ce dernier en possède déjà, et par conséquent autoriser l'unification de la racine d'un arbre avec un nœud quelconque d'un second.

Dans le cadre des grammaires syntagmatiques, les TAG définissent deux opérations de combinaison : l'opération de *substitution*, qui peut être comparée aux opérations de combinaison des grammaires citées ci-dessus en ce qu'elle rattache deux arbres par leurs extrémités, et l'opération d'*adjonction*, qui permet d'insérer un arbre à l'intérieur d'un second (figure 4.30), se rapprochant ainsi de notre opération d'attachement. A l'issue d'une opération d'adjonction, deux nœuds qui étaient auparavent liés dans l'arbre où a lieu l'adjonction, par une relation père-fils peuvent ne plus l'être dans le nouvel arbre, ce qui n'est jamais le cas pour l'opération d'attachement : deux nœuds liés par une dépendance avant une opération d'attachement restent liés après.

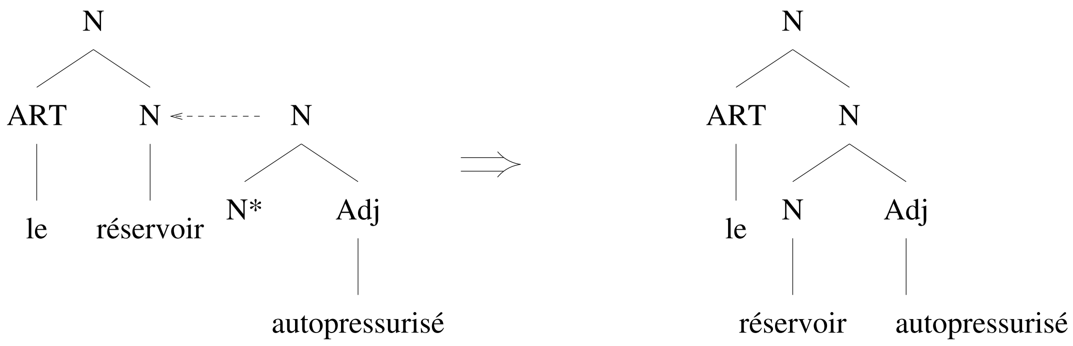


FIG. 4.30: Un exemple d'adjonction dans le formalisme TAG

En ce sens, notre opération est plus proche de l'opération de *furcation* introduite dans les *Segment Grammars* ([de Smedt, 1990]), dont un exemple est donné figure 4.31. La différence avec notre formalisme réside là dans le fait qu'une opération de furcation aboutit toujours à l'augmentation du nombre de fils d'un nœud, du fait que deux arcs ne peuvent s'unifier, ce qui n'est pas le cas de l'opération d'attachement, qui permet l'unification de deux arcs.

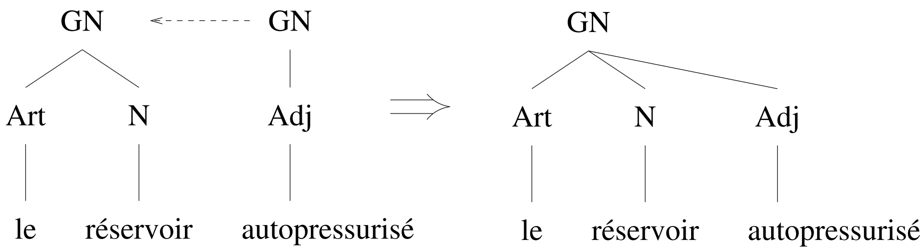


FIG. 4.31: Un exemple de furcation dans le formalisme des *Segment Grammars*

Nous ne pousserons pas plus loin la comparaison de l'opération d'attachement et des opérations de combinaison définies par les grammaires d'arbres adjoints et les *Segment Grammmars*, dans la mesure où la différence des structures manipulées (arbres de dépendances dans notre cas et arbres syntagmatiques dans les deux autres), notamment dans la façon de représenter l'ordre linéaire, fait que la comparaison restera à un niveau très formel et que ses implications en termes plus linguistiques n'apparaîtront pas, à moins d'effectuer une comparaison poussée de ces deux types de représentations syntaxiques.

Chapitre 5

Analyse

L'analyse constitue la première phase de la reformulation. Elle peut être décomposée en deux parties, la construction de la structure syntaxique de surface (*SSyntS*) et la construction d'une structure syntaxique profonde (*SSyntP*) à partir d'une *SSyntS*. La construction de la *SSyntS* constitue la partie la plus complexe du processus de reformulation, sa description va occuper la majeure partie de ce chapitre. Cette complexité s'explique par le nombre de combinaisons devant être pris en compte lors de la construction d'une représentation structurelle d'une phrase et par les phénomènes de non déterminisme que rencontrent cette construction. La construction de la *SSyntP* est, elle, beaucoup plus simple en ce qui concerne les traitements. Elle va par contre être confrontée à un certain nombre de difficultés relevant plus du formalisme que des traitements. Ces problèmes concernent principalement la notion de relation d'abstraction, qui a été définie au chapitre 3, et qui se situe au cœur du processus de construction de la *SSyntP* en établissant un lien entre les *AES* et les *AEP*. Les difficultés de mise en œuvre de cette relation vont être à l'origine d'un accroissement de la complexité du formalisme qui va compter un nouveau type d'élément, les *arbres élémentaires de surface étendus* et du processus de construction de la *SSyntP* qui va comporter une étape supplémentaire.

5.1 Construction de la *SSyntS*

La construction de la *SSyntS* a été présentée au chapitre 3, où quatre étapes distinctes avaient été identifiées : l'analyse morphologique, la recherche de mots composés, l'extraction des *AES* associés aux mots de la phrase à partir d'un dictionnaire syntaxique et, finalement, la composition des *AES* correspondant aux mots de la phrase pour former une ou plusieurs *SSyntS* de cette dernière. Nous ne détaillerons pas ici les trois premières phases, pour nous concentrer sur la dernière, l'analyse syntaxique, qui constitue le cœur du processus de la construction de la *SSyntS*.

L'analyse syntaxique automatique dans le cadre des grammaires de dépendances, que nous appellerons analyse dépendancielle, n'a pas connu le développement spectaculaire de l'analyse dans le cadre des grammaires syntagmatiques (analyse syntagmatique) et en particulier dans le cadre des grammaires indépendantes du contexte. Une des raisons de ce manque de développement réside dans l'absence de consensus sur la représentation formelle de grammaires de dépendances, représentation nécessaire à la tâche d'analyse. Les formalismes élaborés dans les années soixante ([Hays, 1964, Gaifman, 1965, Robinson, 1970]), n'ont pas été retenus par les équipes travaillant sur l'analyse dépendancielle. Une analyse des formalismes cités ci-dessus, telle que celle que nous avons effectuée pour le formalisme de H. Gaifman au chapitre 4, montre en effet que ces derniers sont trop pauvres pour les besoins de modélisation des langues naturelles, poussant les tenants de l'analyse dépendancielle à les modifier ou à en développer de nouveaux, limitant par conséquent la possibilité de bâtir sur des travaux antérieurs. Du fait de la dispersion des efforts de recherche sur l'analyse dépendancielle, cette dernière ne s'est jamais imposée comme discipline à part entière.

On peut s'étonner du fait que les grammaires syntagmatiques et les grammaires hors contextes en particulier, qui ne sont pas plus riches que le modèle de H. Gaifman, aient connu un tel succès, et aient été à l'origine de tant de travaux sur l'analyse syntaxique de la langue naturelle. Cette différence réside dans le fait que les grammaires syntagmatiques, contrairement aux grammaires de dépendances, ont été utilisées pour la formalisation de langages informatiques. C'est dans le cadre des langages informatiques et particulièrement pour des besoins de compilation de programmes que l'analyse syntagmatique a connu un tel développement. Les grammaires de dépendances, qui se sont cantonnées à la représentation syntaxique de la langue naturelle, n'ont pas connu un tel développement.

5.1.1 Quelques analyseurs pour grammaires de dépendances

L'absence de consensus sur une représentation formelle des grammaires de dépendances rend malaisée la comparaison des différents travaux sur l'analyse dépendancielle. Il est en effet impossible de comparer des analyseurs basés sur des formalismes grammaticaux distincts sans prendre en compte leurs différences. Nous allons passer en revue dans les sections suivantes trois travaux sur l'analyse dépendancielle, en décrivant très sommairement certains aspects des formalismes pour lesquels ils ont été conçus. Il ne s'agit pas d'une liste exhaustive, mais d'une sélection de travaux représentant des tendances différentes de l'analyse dépendancielle, décrits dans la littérature. D'autres travaux, que nous ne développerons pas, seront cités ensuite.

Le système PLAIN

Le système PLAIN est à notre connaissance l'un des premiers systèmes d'analyse dépendancielle mentionné dans la littérature. Il a été développé à partir de 1978 à l'Université de Heidelberg par Peter Hellwig. Il est constitué de deux modules : un module d'analyse, prenant en entrée une phrase pour en construire une expression dans un langage appelé DRL (Dependency Representation Language), qui permet de décrire linéairement des arbres syntaxiques de dépendances. Le second module est un module transformationnel appliquant un certain nombre de transformations aux expressions produites par l'analyseur pour produire d'autres expressions en DRL. Les transformations permettent de réaliser, en théorie, certaines inférences, de traduire une expression d'une langue en une autre et de faire de la preuve de théorème. On trouvera une description de l'ensemble du système dans [Hellwig, 1985] et [Hellwig, 1986b].

Le module d'analyse utilise une grammaire lexicalisée (Dependency Unification Grammar) associant à tout mot un ou plusieurs sous-arbres dans lesquels sont décrits tous les dépendants possibles du mot, chaque dépendant étant représenté par une matrice (*slot*). Un premier algorithme d'analyse est décrit dans [Hellwig, 1986a], il est basé sur l'unification des structures élémentaires associées aux mots de la phrase. L'analyseur lit la phrase de gauche à droite, et tente d'unifier la racine de la structure élémentaire associée au mot courant à une matrice de la structure élémentaire de droite ou à une matrice de la structure élémentaire de gauche. L'analyse aboutit lorsque tout mot a été unifié avec une matrice. Une seconde version améliorée de l'analyseur, utilisant une table de sous-chaînes bien formées (*well formed substring table*) et permettant l'analyse de certaines structures non projectives est décrite dans [Hellwig, 1988]. Les détails de l'algorithme ne sont pas décrits dans la littérature : il n'est en particulier pas fait mention de la façon dont est prise en compte l'ambiguïté. L'analyseur a été appliqué à l'allemand, à l'anglais et au français. Il n'existe plus à notre connaissance de travaux sur ce système.

Le système PLAIN possède, malgré son âge, la plupart des caractéristiques des formalismes grammaticaux modernes, telles que la lexicalisation et l'unification. Il offre de plus un mode original de représentation et de prise en compte d'informations sur l'ordre grammatical des mots dans la phrase, sous forme de traits positionnels (voir 4.1.5). L'inconvénient majeur nous semble résider dans la nécessité de décrire dans la grammaire-lexique tous les dépendants possibles de chaque mot au sein de structures élémentaires.

Le système CADET

L'analyseur Cadet a été développé à l'Université de Grenoble par D. Genthial ([Genthial, 1991, Genthial et al., 1990]). Il se distingue des autres travaux en analyse dépendancielle par une représentation des informations syntaxiques sous forme de règles, qui permet de garder une certaine distinction entre le lexique et la grammaire (voir 4.1.2). Les structures de base du langage sont des règles de réécriture dont la partie gauche est constituée d'une liste de catégories et la partie droite, de l'arbre construit à partir de cette liste. Un mécanisme de hiérarchisation décrit dans [Genthial et al., 1990] permet d'utiliser pour des sous-catégories des règles définies au niveau de catégories plus générales. Une règle sujet, par exemple, s'énonce entre un nom et un verbe et peut être utilisée pour des sous-catégories du verbe et du nom. Le système de règles ne permet pas de prendre en compte certains phénomènes, tels que les accords ou les contraintes de sous catégorisation. C'est pourquoi les entrées lexicales sont associées à des structures de traits typés ([Aït-Kaci & Nasr, 1986]) décrivant certaines de leurs caractéristiques morphologiques syntaxiques et sémantiques. Les règles sont, elles, enrichies de conditions sur les structures associées aux éléments combinés. La règle sujet, par exemple, sera conditionnée par le succès de l'unification de la structure de traits associée au nom et de la structure exprimant les contraintes imposées par le verbe à son sujet. La règle sujet se présente de la façon suivante :

$$NomVerbe \rightarrow (Nom)Verbe$$
$$unif(Verbe.syn.suj, Nom)$$

La condition $unif(Verbe.syn.suj, Nom)$ représente la contrainte d'accord entre le verbe et son sujet. Les règles ainsi enrichies permettent de prendre en compte, grâce à l'unification, diverses contraintes morphologiques, syntaxiques et sémantiques.

L'algorithme d'analyse traite la phrase de gauche à droite et construit l'arbre de façon ascendante. Les mots de la phrase sont introduits successivement dans une pile et les règles de la grammaire sont appliquées aux éléments situés à l'extrémité de la pile. L'application d'une règle aux éléments situés à l'extrémité de la pile conduit au remplacement de ces éléments par l'arbre construit grâce à la règle. La complexité en temps de l'analyseur est exponentielle par rapport à la longueur de la phrase analysée.

L'intégration des règles de réécriture et de la représentation des connaissances linguistiques sous forme de structures de traits typés nous semble constituer une solution élégante au problème de la représentation exhaustive de tous les dépendants d'un mot au sein des structures élémentaires, qui apparait dans le formalisme de P. Hellwig. Il n'est en effet plus nécessaire, dans l'approche de D. Genthial de décrire, dans l'entrée lexicale d'un mot, tous ses dépendants possibles, qu'ils soient actants ou circonstants. Le rôle du lexique se résume alors à décrire les caractéristiques propres à un mot, laissant les règles plus générales de syntaxe aux soins de la grammaire. La distinction entre *AES* initiaux et auxiliaires que nous avons présentée en 4.1.4 constitue une autre solution au même problème. Nous avons préféré cette solution pour son homogénéité. Contrairement à D. Genthial, nous n'avons pas de règles de grammaires d'une part et de la descriptions lexicales complexes d'une autre ; toutes les connaissances sont représentées dans les arbres élémentaires sans pour autant tomber dans le problème de description systématique de tous les dépendants d'un mot dans l'arbre élémentaire qui lui est associé. De plus, notre solution permet, par le marquage de certains *AES*, d'interdire des structures syntaxiques correspondant à plusieurs règles utilisées simultanément (voir 4.1.2), caractéristique importante du modèle de reformulation.

Les grammaires de liens

Le formalisme des grammaires de liens (*Link Grammars*) a vu le jour à l'Université de Carnegie-Mellon, en 1991 ([Sleator & Temperley, 1991]). Il ne s'agit pas à proprement parler de grammaires de dépendance, dans la mesure où d'une part, contrairement aux dépendances, les « liens » définis par le formalisme ne sont pas orientés et, d'autre part, des structures cycliques sont possibles. On retrouve par contre beaucoup de notions définies dans le cadre des grammaires de dépendance, telle que la projectivité, rebaptisée planarité et, comme dans la plupart des approches vues ci-dessus, les liens sont étiquetés par des fonctions syntaxiques.

La grammaire est lexicalisée : elle associe à tout mot une structure représentant les liens que le mot peut établir avec d'autres mots de la phrase. Les liens sont regroupés en deux listes, indépendantes des notions de dépendants et de gouverneurs : une liste droite, indiquant les liens que le mot peut établir avec d'autres mots se trouvant à sa droite et, symétriquement, une liste gauche. La conséquence est que les liens sont systématiquement dupliqués, un verbe, par exemple aura un lien gauche de type *sujet* et tout substantif possèdera un lien droit du même type. Dans chaque liste, les liens sont ordonnés selon la distance des mots avec lesquels les liens seront établis. La liste droite d'un verbe, par exemple, comprendra, dans l'ordre, un lien adverbe, un lien objet direct . . . , indiquant que l'adverbe doit se trouver avant l'objet . . . . Nous avons représenté dans la figure 5.1 les structures associées aux mots *le, chat* et *chasse*. L'extrémité des liens est constituée d'une forme géométrique complexe, appelée connecteur. Selon que le connecteur est dirigé vers la droite ou vers la gauche, il s'agit d'un connecteur droit ou d'un connecteur gauche. Pour qu'un lien puisse être établi entre deux mots, un connecteur droit de l'un doit pouvoir s'emboîter dans un connecteur gauche de l'autre.

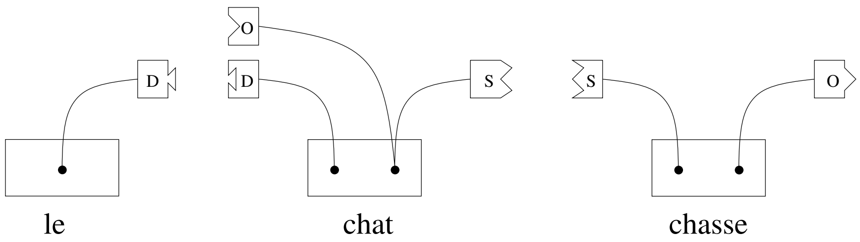


FIG. 5.1: Quelques structures élémentaires des grammaires de liens

L'algorithme d'analyse permet uniquement l'analyse de structures projectives. Sa conception est assez simple : tout mot de la phrase tente d'établir des liens avec d'autres mots de la phrase. L'algorithme est récursif : lorsqu'un lien est établi entre deux mots, l'analyse est relancée sur la portion de phrase située entre les deux mots. La complexité en temps de l'analyse est cubique par rapport à la longueur de la phrase. Un second algorithme d'analyse plus robuste est décrit dans [Grinberg et al., 1995]. Enfin, un modèle probabiliste de grammaires de liens est décrit en [Lafferty et al., 1992]. Une grammaire importante de l'anglais a été implémentée dans ce formalisme, elle a été testée sur des corpus importants (trois millions de mots).

Le formalisme de grammaires de liens ne nous semble pas constituer une évolution importante par rapport à d'autres formalismes, plus anciens, utilisés en analyse dépendancielle. La description grammaticale, qui impose de décrire dans l'ordre tous les liens droits et gauches de tout mot aussi bien pour les gouverneurs que pour les dépendants, est assez coûteuse et donne l'impression d'une certaine redondance. Les algorithmes d'analyse sont par contre plus efficaces, mieux décrits et leurs propriétés sont mieux étudiées que dans d'autres approches, conséquence de la pauvreté du formalisme grammatical.

Les trois approches que nous venons de citer ne représentent qu'une partie des travaux sur l'analyse dépendancielle décrits dans la littérature. On trouvera des approches assez semblables à celles de P. Hellwig dans certains travaux de l'univeristé de Helsinki ([Jäppinen et al., 1987]

) ou dans les travaux de S. Starosta et H. Nomura [Starosta & Nomura, 1986, Starosta, 1975] ou encore dans les analyseurs développés dans le cadre des *Word Grammars* ([Hudson, 1984, Hudson, 1989]) par N. Fraser ([Fraser, 1989, Fraser, 1993]). On pourra aussi mentionner l'analyseur basé sur une architecture multi-agent développé récemment à l'Université de Fribourg ([Bröker et al., 1994b, Bröker et al., 1994a]) et finalement certains travaux de l'Université de Turin [Lesmo & Lombardo, 1992, Lombardo, 1995]. Toutes ces approches présentent des caractéristiques originales ainsi que des points communs. Une analyse précise de leurs points communs et de leurs différences constitue à elle seule un sujet de thèse[1].

5.1.2 Un analyseur pour structures contiguës

Le problème de l'analyse syntaxique d'une phrase P de longueur n peut être vu, d'une façon générale, dans le cadre des grammaires de dépendances, comme l'établissement de $n-1$ dépendances entre les mots de la phrase. Il faut de plus que les $n-1$ dépendances établies entre les mots de la phrase décrivent une structure d'arbres. Pour cela, il faut et il suffit que tout mot de la phrase ait un gouverneur et un seul, à l'exception d'un mot (correspondant à la racine de l'arbre), qui, lui, n'en possède pas. Dans notre cadre, le processus d'analyse consiste à effectuer $n-1$ opérations d'attachement entre les *AES* associés aux mots de la phrase. Une opération d'attachement à droite étant équivalente à l'établissement d'une dépendance droite, et une opération d'attachement à gauche à l'établissement d'une dépendance gauche. La condition sur la structure arborescente du résultat de l'analyse se traduit dans notre cadre par l'interdiction de la cooccurrence de deux opérations $attache(X, a)$ et $attache(Y, a)$ au sein d'une même analyse.

Nous appellerons une séquence de $n-1$ opérations d'attachement aboutissant à une structure d'arbre, une *solution* de l'analyse d'une phrase P de longueur n. Lorsque P est ambiguë, son analyse admet plusieurs solutions. La solution à l'analyse de la phrase de huit mots *le mécanicien installe les faux-tourillons sur le carter*, est donnée dans la figure 5.2[2].

La présentation d'une solution sous la forme d'une suite de $n-1$ opérations d'attachement, comme dans la la figure 5.2, est quelque peu trompeuse. Elle donne l'impression que les $n-1$ opérations d'attachement sont effectuées indépendamment les unes des autres et qu'un *AES* peut participer à plusieurs opérations d'attachement. Ce n'est en réalité pas le cas car à l'issue d'une opération d'attachement réussie entre deux arbres A_1 et A_2, un nouvel arbre A_3 est créé et les deux arbres A_1 et A_2 disparaissent ; ils ne peuvent alors plus être utilisés tels quels dans une autre opération d'attachement. La construction d'une solution ne consiste donc pas à rechercher des séquences de $n-1$ opérations d'attachement puis à les « réaliser » de façon à construire une

[1] Nous n'avons pas cité à dessein dans ce bref tour d'horizon les travaux effectués dans le cadre des grammaires de contraintes ([Karlsson, 1995]) qui relèvent plus de la problématique de l'étiquetage syntaxique que de l'analyse dépendancielle même s'ils s'en rapprochent par certains aspects, tels que l'attribution, dans certains cas, de fonctions syntaxiques aux mots de la phrase.

[2] Pour des raisons de lisibilité, nous n'avons pas représenté des *AES* comme arguments de l'opération d'attachement dans la figure 5.2, mais les ancres des *AES*. Dans une expression Attache-droite(X,Y) ou attache-gauche(X,Y), X représente le gouverneur et Y le dépendant.

1. Attache-gauche (mécanicien, le)
2. Attache-gauche (installe, mécanicien)
3. Attache-gauche (faux-tourillons, les)
4. Attache-droite (installe, faux-tourillons)
5. Attache-droite (installe, sur)
6. Attache-gauche (carter, le)
7. Attache-droite (sur, carter)

FIG. 5.2: Une séquence d'opérations d'attachement

SSyntS, la *SSyntS* est construite au fur et à mesure que les opérations d'attachement à réaliser sont identifiées. La figure 5.2 représente en quelque sorte le résultat de l'analyse mais pas la manière dont elle est effectuée. Celle-ci est représentée dans la figure 5.3 où les opérations d'attachement permettent de construire la *SSyntS* de manière incrémentale. On pourra remarquer que la manière dont est construite la *SSyntS* est fortement influencée par le fait que l'opération d'attachement ne permet de combiner deux arbres qu'à la condition que les segments de phrase qu'ils représentent soient contigus dans la chaîne linéaire.

La construction d'une *SSyntS* ou la recherche d'une solution au problème de l'analyse consiste donc à déterminer les opérations d'attachement à tester ainsi qu'un ordre dans lequel les tester. Elle suppose par conséquent un mécanisme de contrôle. Ce dernier doit permettre d'établir, à l'issue d'une opération d'attachement, quel doit être le prochain attachement à tester, tout en s'assurant que les opérations effectuées jusque là sont compatibles entre elles et que leur réalisation se présente sous la forme d'un arbre.

1. Attache-gauche (mécanicien , le)
   ⟶le mécanicien
2. Attache-gauche (installe , le mécanicien)
   ⟶le mécanicien installe
3. Attache-gauche (faux-tourillons , les)
   ⟶les faux-tourillons
4. Attache-droite (le mécanicien installe , les faux-tourillons)
   ⟶le mécanicien installe les faux-tourillons
5. Attache-droite (le mécanicien installe les faux-tourillons , sur)
   ⟶le mécanicien installe les faux-tourillons sur
6. Attache-gauche (carter , le)
   ⟶le carter
7. Attache-droite (le mécanicien installe les faux-tourillons sur , le carter)
   ⟶le mécanicien installe les faux-tourillons sur le carter

FIG. 5.3: Construction incrémentale d'une solution

L'algorithme

Le contrôle de notre algorithme d'analyse, à l'instar des analyseurs de N. Fraser et de D. Genthial, fait appel à une pile. Cette dernière permet d'une part de déterminer quelles opérations d'attachement effectuer, ainsi que l'ordre dans lequel les effectuer et, d'autre part, de stocker des résultats intermédiaires qui seront nécessaires à un stade ultérieur de l'analyse.

Afin de simplifier la description du fonctionnement de l'analyseur, nous ferons l'hypothèse de la non ambiguïté de la phrase à analyser et des mots la composant, de sorte qu'à un mot ne correspond qu'un *AES* et qu'à la phrase ne correspond qu'une seule *SSyntS*. De plus, nous considèrerons que l'analyseur n'est jamais confronté à des cas d'ambiguïtés locales. Grâce à ces hypothèses, l'analyseur va se comporter de façon déterministe : à tout moment, une seule opération pourra être effectuée. Nous verrons dans la section suivante (5.1.3) comment lever ces hypothèses.

Deux opérations de manipulation de la pile sont définies : une opération d'*empilement*, qui consiste à introduire un *AES* dans la pile et une opération de *réduction*, consistant à tenter l'attachement des deux arbres se trouvant à l'extrémité de la pile. Nous appellerons l'arbre situé à l'extrémité : A_{EXT} et l'arbre se trouvant dans la case précédente de la pile : A_{PREC}. Lors de chaque réduction, deux attachements sont tentés : l'attachement de A_{EXT} dans A_{PREC} et l'attachement de A_{PREC} dans A_{EXT}. Lorsqu'un attachement aboutit, les deux arbres ayant été combinés sont dépilés et le résultat de l'attachement est empilé, réduisant ainsi la taille de la pile d'une unité. Les deux opérations d'empilement et de réduction sont représentées graphiquement dans la figure 5.4 [3].

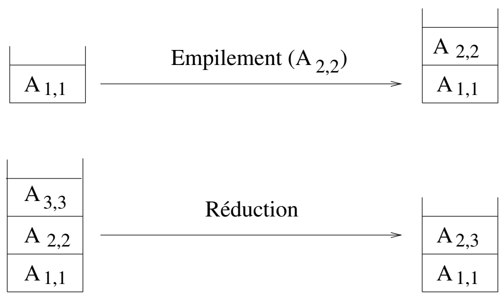


FIG. 5.4: Opérations d'empilement et de réduction élémentaire

L'algorithme d'analyse consiste à lire la phrase de gauche à droite. Pour chaque mot lu, l'*AES* lui correspondant est empilé et une opération de réduction est tentée. Lorsque cette dernière aboutit et qu'un nouvel arbre est créé, une nouvelle réduction de la pile est tentée, et ainsi de suite jusqu'à la première réduction malheureuse. Un nouveau mot est alors lu et le processus est réitéré, jusqu'à atteindre le dernier mot de la phrase. L'algorithme s'arrête donc pour toute phrase de longueur finie. Lorsqu'à l'issue de l'empilement de l'*AES* du dernier mot et après réduction de la pile, cette dernière ne contient qu'un seul arbre, celui-ci constitue le résultat de l'analyse. Au fur et à mesure de la progression de l'analyse, la compatibilité des opérations d'attachement est vérifiée par la construction de l'arbre dans la pile. La présence d'un seul arbre dans la pile à l'issue de l'analyse signifie que toutes les opérations d'attachement effectuées sont compatibles et qu'une solution a donc été trouvée.

Du fait du sens gauche-droite de l'analyse, lorsqu'une opération de réduction est effectuée, l'attachement de l'arbre extrémité (A_{EXT}) de la pile dans son prédécesseur (A_{PREC}) est un attachement à droite, car A_{EXT} correspond à une portion de la phrase située à droite de la portion correspondant à A_{PREC}. Inversement, l'attachement de A_{PREC} dans A_{EXT} est un attachement à gauche. On parlera dans le premier cas de *réduction droite* et dans le second de *réduction gauche*. Une opération de réduction se décompose donc en une réduction droite et une réduction gauche.

L'algorithme est présenté dans la figure 5.5. Un exemple d'analyse est représenté dans la figure 5.6[4].

Au cours de l'analyse, l'arbre syntaxique n'est construit dans aucun des deux sens ascendant ou descendant, mais dans un sens plus difficile à décrire : il s'agit de construire, en partant du premier

[3]Les indices apparaissant au niveau des arbres représentent le segment de la phrase auquel correspond l'arbre ou l'extension de sa racine. L'arbre $A_{2,5}$, par exemple, correspond au segment $m_2\, m_3\, m_4\, m_5$. Les arbres élémentaires, correspondant à un seul mot, seront représentés sous la forme $A_{i,i}$.

[4]Pour des raisons de lisibilité, nous n'avons pas représenté d'arbres dans la pile mais les segments de phrase correspondant aux arbres.

Entrées: Une liste de N AES : $AES_1 \ldots AES_N$
Une pile P

1) Pour i allant de 1 à N
faire
  2) Empile(AES_i)
  3) h $\longleftarrow$ hauteur(P)
  4) Réduire-droite(P)
  5) Réduire-gauche(P)
  6) Si hauteur(P) $<$ h Alors 3
fin faire
7) Si hauteur(P) $=$ 1 Alors succès
Sinon échec

FIG. 5.5: L'algorithme d'analyse

mot de la phrase, un arbre correspondant au segment de phrase le plus long possible. Lorsque la construction est bloquée, l'arbre construit jusque là est stocké dans la pile et le processus est réitéré à partir du premier mot suivant le segment déjà reconnu. Si ce nouveau processus aboutit à la construction d'un nouvel arbre alors l'attachement de ce dernier avec l'arbre préalablement stocké dans la pile est tenté. C'est ainsi que dans notre exemple le sous-arbre correspondant à la séquence *le mécanicien vérifie* est d'abord construit. Un problème apparaît à l'occasion de l'empilement de l'article *la*. Il n'est en effet pas possible de construire un arbre correspondant à la séquence *le mécanicien vérifie la*. C'est là que les capacités de stockage de la pile sont exploitées, l'arbre correspondant à la séquence *le mécanicien vérifie* est stocké et la construction d'un nouvel arbre est entreprise à partir de l'article *la* et aboutira à l'arbre *la troisième pompe*. Un nouvel arbre correspondant à *le mécanicien vérifie la troisième pompe* est alors construit. La situation est résumée dans la figure 5.7 où nous avons représenté au-dessous de la phrase les segments reconnus au fur et à mesure de l'analyse. A chaque étape, les segments représentés en gras sont regroupés par une opération d'attachement et apparaissent sous la forme d'un seul segment à l'étape suivante.

La première question qui se pose à la suite de la description de l'ordre de construction de l'arbre est de savoir si tout arbre ordonné peut être construit suivant cette stratégie. La réponse est négative : certains arbres ne peuvent être construits ainsi, comme le montre l'exemple de la figure 5.8, où aucun couple de mots liés par une dépendance n'occupe deux cases adjacentes de la pile. Ainsi, tous les *AES* des mots de la phrase sont empilés, mais aucune réduction ne peut avoir lieu. Nous parlerons dans ce cas de *structures non contiguës.*

L'impossibilité d'analyser des phrases non contiguës découle directement du fait que l'opération d'attachement permet uniquement de regrouper des segments de phrase contigus. En relâchant cette contrainte, et en autorisant l'attachement d'arbres situés dans des cases non contiguës de la

Fig. 5.6: Un exemple d'analyse

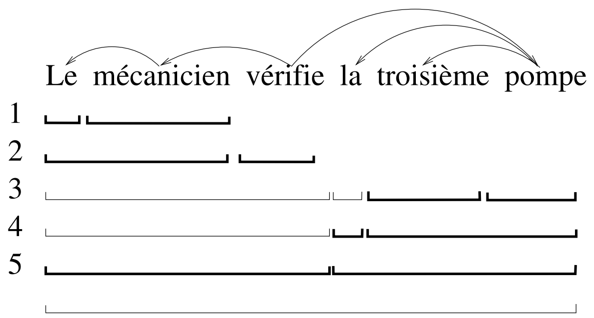


Fig. 5.7: Etapes de l'analyse d'une phrase

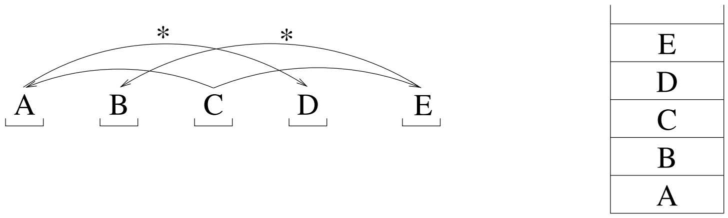


FIG. 5.8: Une structure non contiguë et la configuration de pile lui correpondant

pile, la pile de la figure 5.8 pourrait être réduite et l'analyse menée à bien. Un tel relâchement provoquerait par contre une augmentation du nombre d'opérations d'attachement testées au cours de l'analyse, dégradant ainsi les performances de l'analyseur.

Bien que l'exemple de la figure 5.8 contienne deux dépendances non projectives marquées chacune d'une astérisque, les conditions de contiguïté et de projectivité ne se confondent pas : une structure contiguë peut être non projective, comme le montre la structure contiguë non projective de la figure 5.9[5].

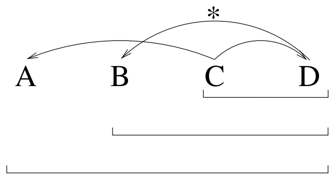


FIG. 5.9: Une structure contiguë non projective

Par contre, toute structure projective est contiguë, la projectivité est un cas particulier de contiguïté. De même, la contiguïté ne se confond pas avec la pseudo-projectivité et certaines structures pseudo-projectives ne sont pas contiguës. Cette situation peut-être problématique dans la mesure où certaines structures pseudo-projectives ne peuvent être analysées, car non contiguës. Nous avons observé en pratique que les exemples linguistiques pseudo-projectifs recensés sont contigus, mais une nouvelle propriété, caractérisant les structures pseudo-projectives contiguës, qui satisfasse donc à la fois aux besoins de description des structures linguistiques et aux contraintes d'analyse reste à définir. L'impossibilité d'analyser des structures non contiguës ne nous apparaît pas comme une contrainte dans le cadre du TAL dans la mesure où ces dernières semblent improbables en linguistiques.

L'utilisation d'une pile dans l'analyseur présente en fin de compte trois avantages. Elle permet un contrôle simple de l'analyseur, réduit le nombre d'opérations d'attachement à tester pendant

[5]On peut reconnaître une structure contiguë en dessinant sous cette dernière les segments regroupés à chaque étape de l'analyse, comme dans la figure 5.7. S'il est possible de regrouper tous les mots au sein d'un même segment en ne regroupant à chaque étape que des segments contigus alors la structure est contiguë.

l'analyse et n'exclut de l'analyse que des structures linguistiquement improbables : les structures non contiguës.

SSyntS **et** *SSyntS Analytique*

A l'issue de l'analyse d'une phrase P de longueur n, deux structures sont produites : une *SSyntS* et les $n-1$ opérations d'unification permettant de construire la *SSyntS*. Cette suite d'opérations constitue la *solution* dont il était question en début de section, elle peut être représentée graphiquement sous la forme d'un arbre dont les *AES* constituent les nœuds et dont les branches (représentées en pointillés) relient les nœuds des *AES* devant être unifiés pour construire la *SSyntS*. Cette structure correspond aussi à la *SSyntS Analytique* que nous avions introduite au chapitre 3 et qui avait été comparée aux arbres de dérivation dans le formalisme TAG. Les deux structures sont représentées graphiquement dans la figure 5.10.

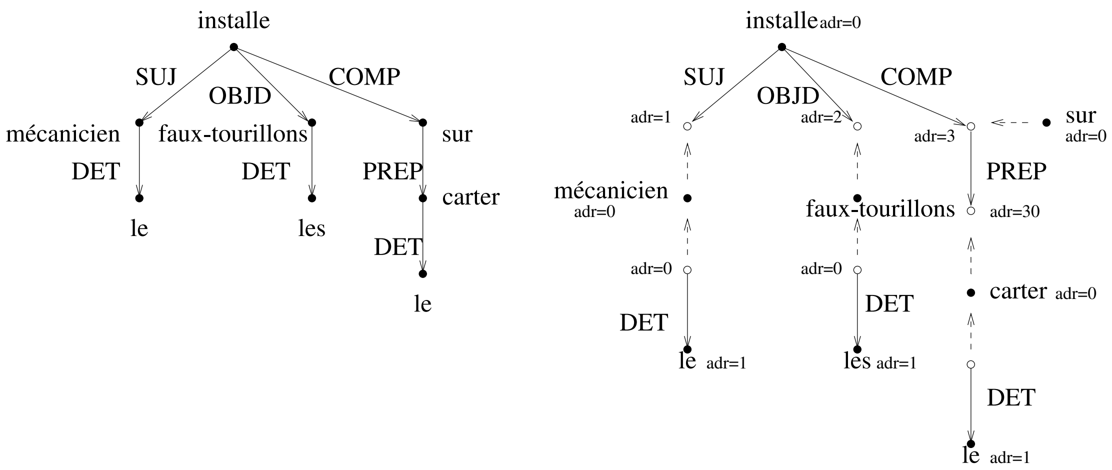


FIG. 5.10: Une *SSyntS* et une *SSyntS Analytique*

Les nœuds des *AES* apparaissant dans la *SSyntS Analytique* de la figure 5.10 ont été enrichis d'un trait *adr*, qui associe à tout nœud un nombre que nous appellerons l'*adresse* du nœud. Celle-ci est unique dans le contexte d'un *AES* particulier, elle permet par conséquent d'identifier un nœud unique de l'*AES* [6]. La notion d'adresse de nœud permet une représentation linéaire de la *SSyntS Analytique* sous forme d'une séquence d'opérations d'unification. La *SSyntS Analytique* de la figure 5.10 a été représentée linéairement dans la figure 5.11. Cette représentation est importante car c'est elle qui sera utilisée dans la suite du processus de reformulation. Elle est plus riche que la *SSyntS* dans la mesure où elle indique comment construire cette dernière à partir des *AES*.

[6]Le nœud d'adresse X d'un *AES* sera représenté par l'expression *AES*@X.

$$
\begin{array}{l}
\text{Unif}(AES_{mécanicien}@0\ ,\ AES_{le}@0)\\
\text{Unif}(AES_{installe}@1\ ,\ AES_{mécanicien}@0)\\
\text{Unif}(AES_{faux-tourillons}@0\ ,\ AES_{les}@0)\\
\text{Unif}(AES_{installe}@2\ ,\ AES_{faux-tourillons}@0)\\
\text{Unif}(AES_{installe}@3\ ,\ AES_{sur}@0)\\
\text{Unif}(AES_{carter}@0\ ,\ AES_{le}@0)\\
\text{Unif}(AES_{installe}@30\ ,\ AES_{carter}@0)
\end{array}
$$

FIG. 5.11: Représentation linéaire d'une *SSyntS Analytique*

5.1.3 Gestion du non déterminisme

Les hypothèses que nous avons effectuées dans la section précédente concernant la non ambiguïté de la grammaire et des phrases traitées conféraient à l'algorithme un comportement déterministe. A l'issue de toute opération d'empilement ou de réduction, une seule opération était possible : après un empilement, une réduction ; après une réduction réussie, une nouvelle réduction et après une réduction malheureuse, un nouvel empilement. La prise en compte de l'ambiguïté va introduire des cas de non déterminisme dans l'analyse. Deux types d'ambiguïtés peuvent se présenter : des ambiguïtés lexicales où à un mot correspondent plusieurs *AES* (il faut alors savoir lequel empiler) et des ambiguïtés structurales où, à l'issue d'une opération de réduction, plusieurs arbres sont créés (il faut alors savoir lequel garder dans la pile). La création de plus d'un arbre à l'issue d'une réduction peut, elle, avoir deux causes : le succès des deux opérations d'attachement lors d'une réduction ou la création de plusieurs arbres à l'issue d'un attachement.

La distinction des cas d'ambiguïté que nous avons effectuée (plusieurs *AES* correspondant à un mot et la création de plusieurs arbres à l'issue d'une opération de réduction) ne se confond pas exactement avec la distinction classique entre ambiguïté lexicale et ambiguïté structurale. Ceci est dû à l'intégration d'informations lexicales et syntaxiques au sein d'*AES*. La notion d'*AES* est en effet plus riche que la notion de partie de discours classique, du fait qu'elle représente explicitement des informations sur le contexte syntaxique d'un mot. Ainsi, à titre d'exemple, la grammaire permet de distinguer des *AES* associés à une préposition selon que cette dernière dépend d'un nom ou d'un verbe, différence qui n'est pas reflétée dans le nom de la catégorie *Prep*. Dans un tel cas, une ambiguïté de rattachement prépositionnel où la préposition peut être rattachée à un verbe ou à un nom sera vue comme une ambiguïté « lexicale », du fait qu'à la préposition correspondent deux *AES*, et non comme une ambiguïté structurale (voir figure 5.13). Nous garderons l'expression d'ambiguïté lexicale pour les cas où à un mot correspondent plusieurs *AES* et nous parlerons d'ambiguïté de rattachement dans le second cas.

Ces deux cas de non déterminisme ne peuvent être traités localement : on ne peut décider au moment d'un empilement quel *AES* choisir parmi plusieurs et l'on ne peut décider lorsque plusieurs arbres sont créés à l'issue d'une opération de réduction, lequel est le bon. De plus, certaines phrases peuvent être globalement ambiguës, auquel cas toutes les solutions devront être trouvées, ce que ne permet pas l'analyseur tel qu'il a été décrit. C'est pourquoi nous allons introduire dans les deux sections suivantes des mécanismes de gestion du non déterminisme.

Duplication de piles

Une solution possible pour faire face au non déterminisme consiste à dupliquer les piles lors de chaque cas de non déterminisme, qu'il soit provoqué par une ambiguïté lexicale ou une ambiguïté de rattachement. Dans les cas d'ambiguïté lexicale, où à un mot correspondent plusieurs *AES*, la pile est dupliquée en autant d'exemplaires que le mot possède d'*AES* et chaque *AES* est empilé dans une pile. Dans les cas d'ambiguïté de rattachement, lorsqu'une opération de réduction aboutit à la création de N arbres, la pile est dupliquée N fois, et chaque arbre est placé à l'extrémité d'une pile. Ces deux cas sont représentés graphiquement dans la figure 5.12.

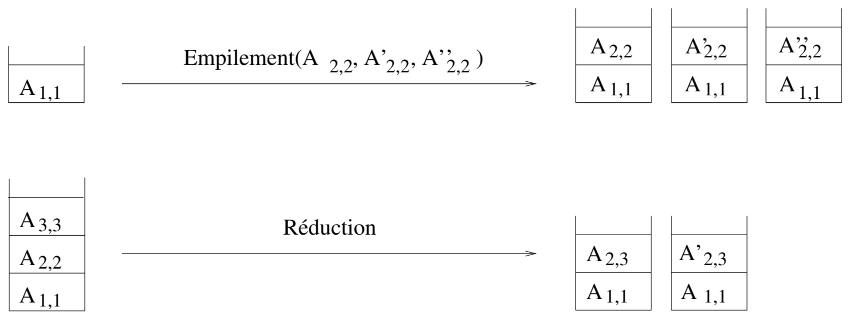


FIG. 5.12: Duplication de piles dans des cas de non déterminisme

Moyennant les modifications des deux opérations d'empilement et de réduction, le principe de l'algorithme d'analyse reste inchangé, l'analyse non déterministe peut être vue comme plusieurs analyses déterministes menées en parallèle. A l'issue de l'analyse, plusieurs piles auront été créées. Celles qui parmi ces dernières ne contiennent qu'un seul arbre représentent les analyse réussies et ces arbres constituent les *SSyntS* correspondant à la phrase. Un exemple abrégé d'analyse de la phrase ambiguë *Jean vit un homme avec un télescope* est représenté dans la figure 5.13. Il s'agit là d'une ambiguïté lexicale : la préposition *avec* est associée à deux *AES*, un *AES* correspondant à un complément prépositionnel nominal et un *AES* correspondant à un complément prépositionnel verbal. Lors de l'empilement de la préposition, la pile est dupliquée et chaque *AES* est empilé dans un exemplaire de la pile.

La prise en compte du non déterminisme par duplication de piles tel que nous venons de la décrire pose des problèmes de complexité en temps de l'algorithme d'analyse. En effet, le nombre de piles créées durant un traitement augmente exponentiellement par rapport au nombre de mots traités : lors de chaque empilement des *AES* associés à un mot m_i, le nombre de piles est

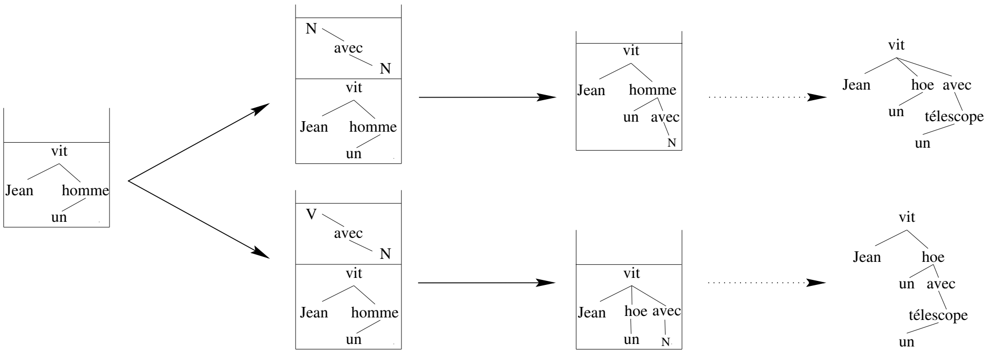


FIG. 5.13: Analyse d'une phrase ambiguë

multiplié par le nombre d'*AES* associés à m_i, provoquant une explosion combinatoire exponentielle. Nous allons proposer dans la section suivante une gestion plus efficace du non déterminisme.

Introduction d'une pile-graphe

Dans le traitement du non déterminisme proposé dans la section précédente, les piles créées à l'issue d'une duplication sont totalement indépendantes et, dans certains cas, des opérations identiques d'attachement sont effectuées dans plusieurs d'entre elles. La figure 5.14 illustre par un exemple un tel cas où, du fait que les *AES* $A_{2,2}$ et $A_{3,3}$ sont représentés dans deux piles différentes, les opérations d'attachement $Attache(A_{2,2}, A_{3,3})$ et $Attache(A_{3,3}, A_{2,2})$ sont effectuées deux fois lors de la réduction des piles.

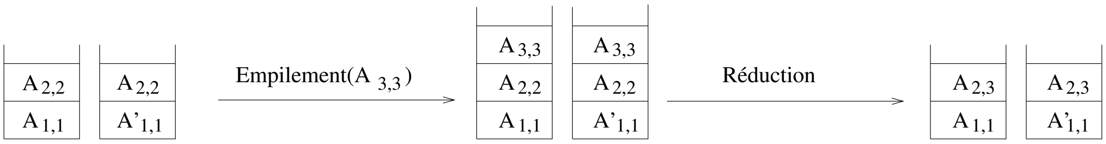


FIG. 5.14: Duplication d'opérations d'attachement

Dans le but de contenir l'explosion combinatoire que nous avons mise à jour dans la section précédente, nous allons factoriser un certain nombre d'opérations qui étaient effectuées plusieurs fois, dans plusieurs piles. Cette factorisation des opérations va être rendue possible par une factorisation des piles au sein d'une pile à structure de graphe (*pile-graphe* par la suite), introduite dans [Tomita, 1987] et [Tomita, 1985] et adaptée à plusieurs formalismes syntaxiques dans [Tomita, 1988]. Le but d'une pile-graphe est précisément d'éliminer la duplication de certaines opérations identiques apparaissant lors de processus non déterministes.

Une pile-graphe peut être vue comme la factorisation de plusieurs piles comportant certains éléments en commun, comme le montre la figure 5.15 pour deux piles possèdant en commun les éléments $A_{2,2}$ et $A_{3,3}$. L'introduction d'une pile-graphe dans l'analyseur ne va pas modifier le fonctionnement de ce dernier : seules seront modifiées les deux opérations d'empilement et de réduction.

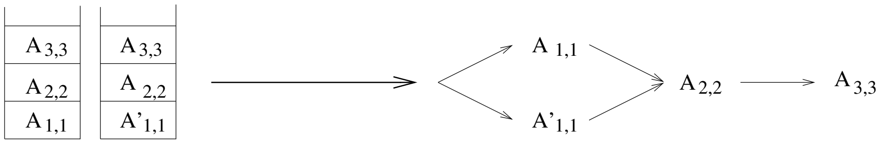


FIG. 5.15: factorisation de plusieurs piles sous forme d'une pile-graphe

L'empilement de n AES va consister à créér n nouvelles extrémités à la pile-graphe (une extrémité pour chaque AES) et à relier chacune de ces n extrémités à toutes les extrémités précédentes de la pile-graphe, comme l'illustre la figure 5.16, où l'empilement des deux AES $A_{2,2}$ et $A'_{2,2}$ provoque la création de six arcs liant les anciennes extrémités de la pile-graphe aux nouvelles. A l'issue de l'opération d'empilement, la pile-graphe compte autant d'extrémités que d'arbres élémentaires y ont été empilés.

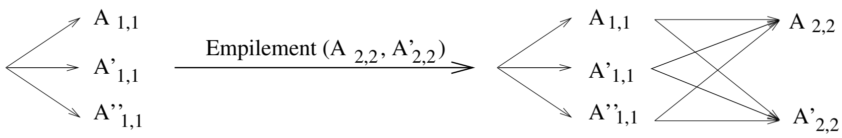


FIG. 5.16: Empilement de deux AES dans une pile-graphe

La réduction de la pile-graphe consiste à essayer les attachements de chaque extrémité avec tous ses prédécesseurs. Dans l'exemple de la figure 5.17, les attachements de $A_{i,j}$ avec respectivement $A_{j+1,j+1}$ et $A'_{j+1,j+1}$ sont effectués. L'attachement de $A_{i,j}$ et de $A_{j+1,j+1}$ donne naissance à deux arbres, $A_{i,j+1}$ et $A'_{i,j+1}$, provoquant une duplication de l'extrémité précédemment constituée par $A_{j+1,j+1}$. La seconde opération d'attachement, entre $A_{i,j}$ et $A'_{j+1,j+1}$ n'aboutit pas, laissant l'extrémité formée par $A'_{i,j+1}$ inchangée. Ainsi, à la faveur de l'opération de réduction, le nombre d'extrémités de la pile-graphe augmente d'une unité, passant de deux à trois.

La duplication d'opérations d'attachement que nous avions observée ci-dessus ne va plus se produire avec l'utilisation de la pile-graphe. Nous avons repris, dans la figure 5.18, l'exemple à l'aide duquel nous avions illustré la duplication d'opérations similaires. Cette duplication n'a maintenant plus lieu, les opérations $Attache(A_{2,2}, A_{3,3})$ et $Attache(A_{3,3}, A_{2,2})$ ne sont plus effectuées qu'une seule fois, conséquence du fait que $A_{2,2}$ et $A_{3,3}$ ne sont plus représentés qu'une fois.

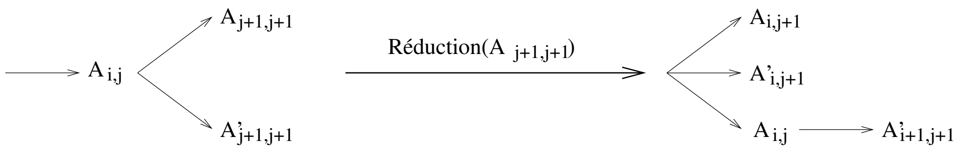


FIG. 5.17: Réduction d'une pile-graphe

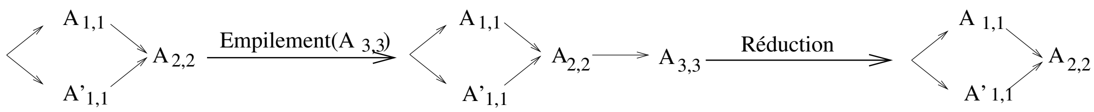


FIG. 5.18: Elimination de la duplication d'opérations d'attachement grâce à l'utilisation d'une pile-graphe

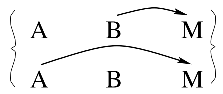
Il faut toutefois noter que le recours à une pile-graphe n'élimine pas tous les cas de duplication d'opérations identiques. En ce qui concerne les performances de l'analyseur, l'utilisation de la pile-graphe assure une complexité polynomiale en temps de l'algorithme d'analyse.

Analyse bidirectionnelle

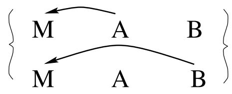
Le mode de gestion du non déterminisme que nous avons décrit ci-dessus, qu'il soit réalisé par duplication de pile ou grâce à une pile-graphe, ne permet pas de traiter tous les cas d'ambiguïté. Dans certains cas, où plusieurs analyses sont possibles, une seule sera reconnue par l'analyseur. Un tel cas se présente, par exemple, lors du traitement d'une séquence $A\,M\,B$, lorsque M peut être gouverné dans une première configuration par A et dans une seconde par B. Dans ce cas d'ambiguïté, seule la solution où A gouverne M (établissement d'une dépendance droite) sera construite. Le déroulement de l'analyse permet de mieux voir le problème : à l'issue des empilements de A et de M, l'attachement de A et M est tenté et la dépendance droite $A \longrightarrow M$ est créée, empêchant ainsi l'attachement de M et de B lorsque ce dernier sera empilé et la pile réduite. En effet, M ayant A pour gouverneur, il ne pourra être rattaché à B, ce qui empêche de construire la seconde solution.

Le problème ne se pose pas dans les cas d'ambiguïté où deux gouverneurs possibles d'un mot M se trouvent à sa gauche, comme dans les cas de rattachements prépositionnels ambigus. Ainsi, dans la configuration $A\,B\,M$ où M peut être gouverné par A ou B, les deux solutions seront trouvées. C'est dans les cas où les deux gouverneurs possibles d'un mot se trouvent de part et d'autre de ce dernier ou tous deux à sa droite que le problème apparaît. On peut donc distinguer trois types d'ambiguïté :

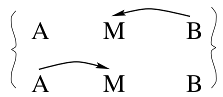
- Ambiguïté gauche : les différents gouverneurs possibles d'un mot M se trouvent à sa gauche. Dans ce cas, les différentes analyses sont détectées par l'analyseur.



- Ambiguïté droite : les différents gouverneurs possibles d'un mot M se trouvent à sa droite. Dans ce cas, seule la solution où M est rattaché à son gouverneur le plus proche (A dans le schéma) est détectée.



- Ambiguïté bilatérale : un mot M peut avoir des gouverneurs à droite ou à gauche. Dans ce cas, seuls les attachements de M avec les gouverneurs situés à gauche (A dans la schéma) sont détectés.



Ce problème de non reconnaissance de certaines ambiguïtés est dû au comportement « impatient » de l'analyseur qui tente d'effectuer un attachement dès que possible. Lorsque l'attachement aboutit, la situation qui prévalait avant l'attachement est oubliée, ce qui empêche, la construction de certaines solutions. Il est difficile de prévoir, étant donnée une grammaire particulière, si de tels cas d'ambiguïté peuvent apparaître, car ces derniers ne doivent pas forcément correspondre à des cas d'ambiguïté globale, mais peuvent correspondre à des ambiguïtés locales apparaissant lors du traitement. Dans la pratique, nous avons observé un tel cas lors de l'analyse de structures coordonnées.

On peut remarquer que l'analyseur se comporte de façon assymétrique : il favorise systématiquement les attachements à droite (établissement de dépendances droites). Ce comportement est une conséquence du sens de l'analyse. En effet, lorsqu'un *AES* correspondant à un mot M est introduit dans la pile, l'analyseur tente de le rattacher, par un attachement à droite, à la partie de la phrase déjà analysée : la partie située à *gauche* de M. C'est uniquement lorsque l'*AES* correspondant à M ne peut être rattaché par un attachement à droite, qu'un nouveau mot est empilé et qu'un attachement à gauche est tenté. Lorsque l'analyse est effectuée de droite à gauche, on observe un comportement symétrique : lorsqu'un *AES* correspondant à un mot M est introduit dans la pile, l'analyseur tente de le rattacher, par un attachement à gauche, à la partie de la phrase déjà analysée, située à *droite* de M. C'est uniquement lorsque l'*AES* correspondant à M ne peut être rattaché par un attachement à gauche qu'un nouveau mot est empilé et qu'un attachement à droite est tenté. Ainsi, lorsque l'analyse est effectuée de droite à gauche, dans les cas d'ambiguïté droite, toutes les analyses sont détectées ; dans les cas d'ambiguïté gauche, une seule analyse est détectée ; enfin, dans les cas d'ambiguïtés bilatérales, seules les analyses correspondant à des attachements à gauche sont détectées (voir la figure 5.19).

Cette assymétrie va permettre de résoudre le problème des ambiguïtés non détectées. Il suffit, en effet, d'analyser la phrase en deux passes : une première passe de gauche à droite et une seconde de droite à gauche. L'adaptation de l'algorithme à l'analyse droite-gauche est minime : elle consiste à modifier les opérations de réduction droite et de réduction gauche. La réduction droite, qui consistait à attacher l'arbre situé à l'extrémité de la pile (A_{EXT}) dans l'arbre situé dans la case adjacente (A_{PREC}) consistera maintenant à attacher A_{PREC} dans A_{EXT}. La modification symétrique sera appliquée à l'opération de réduction gauche. Et, bien entendu, l'ordre de parcours de la phrase doit être inversé.

A l'issue des deux passes, certains arbres auront été créés en deux exemplaires et l'un des deux devra être éliminé. Nous avons repris dans la figure 5.19 les différents cas d'ambiguïté et les cas décelés, selon que l'analyse est effectuée de gauche à droite ou de droite à gauche. La figure 5.19 montre que c'est dans les cas d'ambiguïté gauche et d'ambiguïté droite que des solutions sont construites deux fois.

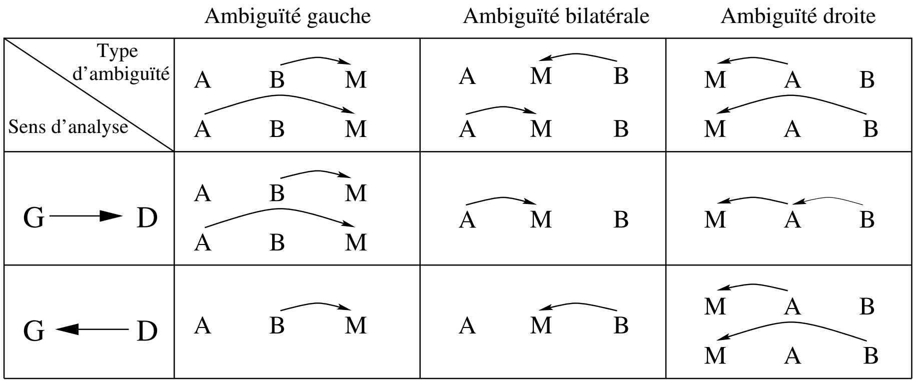


FIG. 5.19: Types d'ambiguïté décelés selon le sens de l'analyse

5.2 Construction de la *SSyntP*

Le principe de construction de la *SSyntP* a été brièvement évoqué au chapitre 3. Il consiste à remplacer dans la *SSyntS Analytique*, issue de la première phase de l'analyse, les *AES* entrant dans sa composition, par des *AEP* leur correspondant par relation d'abstraction. Le remplacement des *AES* par des *AEP* va donner naissance à une *SSyntP Analytique*. L'exemple du chapitre 3 a été repris dans la figure 5.20, où les *AES* entrant dans la composition de la *SSyntS Analytique* ont été remplacés par des *AEP*. Certains *AES*, tels que les *AES* associés aux déterminants ont disparu, ils ont été remplacés au niveau syntaxique profond par des traits. De même, la préposition régie *de* a disparu.

La construction de la *SSyntP* peut être décomposée en deux étapes, une étape de création des *AEP* correspondant aux *AES* et une étape de substitution des *AEP* nouvellement créés aux

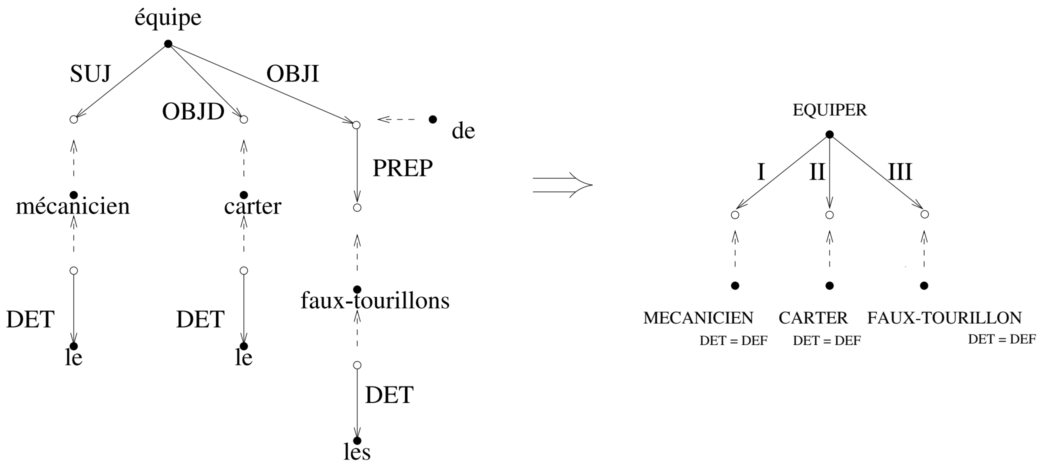


FIG. 5.20: Construction d'une *SSyntP Analytique* à partir d'une *SSyntS Analytique*

AES. Les relations entre *AES* et *AEP* sont établies à l'aide des règles d'abstraction, introduites au chapitre 3, où elles étaient présentées de façon statique, comme un lien entre un *AES* et un *AEP* existants. Nous les présenterons ici d'un point de vue dynamique comme moyen de *créér* un *AEP* à partir d'un *AES* existant.

Il est important de remarquer que le processus de construction d'une *SSyntP* à partir d'une *SSyntS*, que nous venons d'évoquer, ne peut échouer. Il n'y a plus à *contrôler*, à ce niveau, que la *SSyntP* produite est bien formée. Cette garantie de bonne formation de la *SSyntP* produite est le résultat de l'intégration, décrite en 3.2, d'informations de niveau syntaxique profond (tel que le schéma de régime des lexèmes) dans les *AES* et leur prise en compte dès la construction de la *SSyntS*. Cette prise en considération anticipée de contraintes de niveau syntaxique profond lors de la construction de la *SSyntS* a pour conséquence de décharger le processus de construction de la *SSyntP* de tout souci de cohérence de la structure engendrée. Il s'ensuit une relative simplicité du processus de construction de la *SSyntP*.

5.2.1 Les règles d'abstraction

Les règles d'abstraction établissent une relation entre un type d'*AES* et un type d'*AEP*. Elles sont représentées de la façon suivante[7] :

$$\mathcal{AES} \Rightarrow \mathcal{AEP} \mid f$$

où $\mathcal{AES}$ et $\mathcal{AEP}$ sont respectivement un type d'*AES* et un type d'*AEP*, et f une fonction partielle bijective, que nous appellerons *fonction de correspondance*, définie de l'ensemble des

[7] Cette façon de représenter les règles d'abstraction est empruntée aux dendrogrammaires de [Gladkij & Mel'čuk, 1975] présentées en 3.5

nœuds de l'$\mathcal{AES}$ (identifiés par leur adresse) vers l'ensemble des nœuds de l'$\mathcal{AEP}$. C'est cette fonction, grâce aux liens qu'elle établit entre les nœuds des arbres élémentaires mis en jeu dans la règle, qui permet de substituer un arbre élémentaire à un autre au sein d'une structure analytique.

Les règles d'abstraction représentent d'une part une correspondance entre un type d'*AES* et un type d'*AEP* et permettent d'autre part d'établir les liens entre les nœuds des deux arbres élémentaires nécessaires à l'opération de substitution. C'est donc au niveau des *types* d'*AES* et d'*AEP* que sont définies les règles d'abstraction. Elles s'appliquent par contre sur des instances d'*AES* pour générer des instances d'*AEP*. En sus de leur aspect stuctural, les règles d'abstraction recouvrent un aspect lexical, qui permet de transformer un lexème de surface (l'ancre lexicale de l'*AES*) en un lexème profond (l'ancre lexicale de l'*AEP*). Cette transformation ne prend en charge que les cas simples de correspondance entre lexèmes de surface et lexèmes profonds, élimination de certains traits morphologiques et ajout d'autres traits. Les cas plus complexes de reconnaissance de phénomènes de cooccurrences lexicales restreintes seront traités lors de la transformation de la *SSyntP*, décrite au chapitre 6.

La règle d'abstraction d'un $\mathcal{AES}$ associé à un verbe trivalent est représentée graphiquement dans la figure 5.21. Les liens entre nœuds établis par la fonction de correspondance ont été représentés par des doubles flèches en pointillés reliant les nœuds de l'$\mathcal{AES}$ aux nœuds de l'$\mathcal{AEP}$. Ces liens entre nœuds ne seront pas systématiquement représentés graphiquement dans les schémas, afin de ne pas surcharger ces derniers.

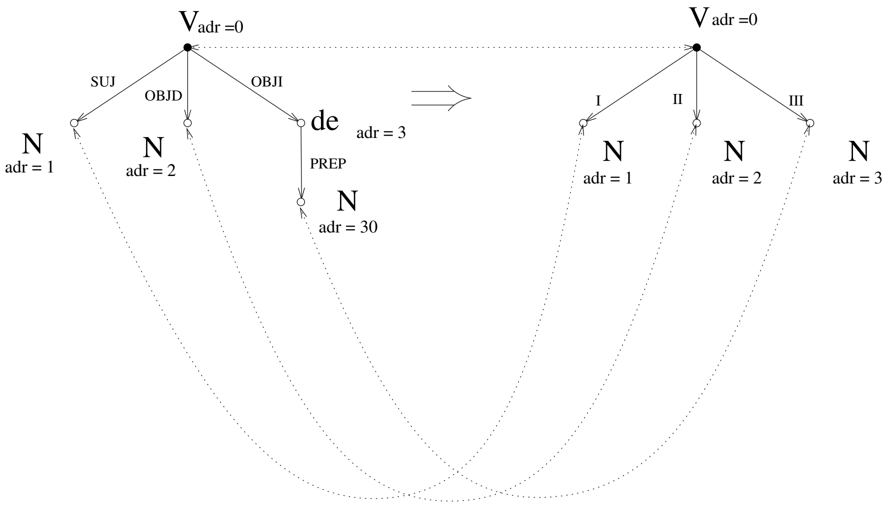


FIG. 5.21: Représentation graphique d'une règle d'abstraction

La règle de la figure 5.21 sera représentée dans notre formalisme de la façon suivante :

$$v_{n_1 n_2 de n_3} \Rightarrow V_{N_I N_{II} N_{III}} \mid f$$

où $v_{n_1 n_2 den_3}$ est un type d'*AES* associé à un verbe trivalent et $V_{N_I N_{II} N_{III}}$ le type d'*AEP* qui lui correspond. f est définie de la façon suivante[8] :

$$f(0) = 0\ ,\ f(1) = 1\ ,\ f(2) = 2\ ,\ f(30) = 3$$

L'application d'une telle règle à un *AES* du verbe *équiper* est représentée dans la figure 5.22.

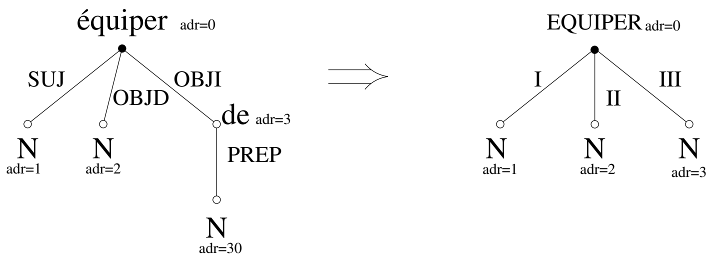


FIG. 5.22: Un *AES* lié à un *AEP* par relation d'abstraction

La fonction de correspondance permet d'établir les liens suivants[9] :

$$\begin{array}{c} f(AES_{\acute{e}quiper}@0) = AEP_{EQUIPER}@0 \\ f(AES_{\acute{e}quiper}@1) = AEP_{EQUIPER}@1 \\ f(AES_{\acute{e}quiper}@2) = AEP_{EQUIPER}@2 \\ f(AES_{\acute{e}quiper}@30) = AEP_{EQUIPER}@3 \end{array}$$

5.2.2 Remplacement des *AES* par des *AEP*

La construction d'une *SSyntP* va consister à remplacer, dans la *SSyntS Analytique*, les *AES* entrant dans sa composition par les *AEP* leur correspondant par relation d'abstraction, constituant ainsi, au fil des substitutions, une *SSyntP Analytique*. La mise en œuvre de l'opération de substitution est simple : lorsque deux nœuds d'*AES* participent à une opération d'unification dans la *SSyntS Analytique*, leurs images respectives par les fonctions de correspondance, associées aux règles d'abstraction, devront être unifiées dans la *SSyntP Analytique*[10] :

$$Unif(AES_1@\alpha,\ AES_2@\beta)\ \Rightarrow\ Unif(f_1(AES_1@\alpha),\ f_2(AES_2@\beta))$$

[8] On pourra remarquer que le nœud d'adresse 3, correspondant à la préposition *de*, n'a pas d'image par f.

[9] Par souci de lisibilité, nous représenterons l'*AES* associé à un mot m par l'expression AES_m. L'*AES* associé au verbe *équiper*, par exemple, sera représenté par l'expression $AES_{\acute{e}quiper}$ au lieu de $vn_1n_2den_3(\acute{e}quiper)$, selon les conventions proposées au chapitre 3. La même notation abrégée sera utilisée pour les *AEP*.

[10] f_1 est la fonction de correspondance de la règle d'abstraction appliquée à AES_1 et f_2, celle de la règle appliquée à AES_2.

Dans l'exemple de la figure 5.20, l'unification des nœuds d'*AEP* correspondant à MÉCANICIEN et à ÉQUIPE, par exemple, est déduite de l'unification des nœuds d'*AES* de *mécanicien* et de *équipe* :

$$\begin{aligned} Unif(AES_{\acute{e}quipe}@1,\ & AES_{m\acute{e}canicien}@0) \\ & \Rightarrow Unif(f(AES_{\acute{e}quipe}@1),\ f(AES_{m\acute{e}canicien}@0)) \\ & \Rightarrow Unif(AEP_{EQUIPER}@1,\ AEP_{MECANICIEN}@0) \end{aligned}$$

L'importance de la condition de bijectivité de la fonction de correspondance que nous avions imposée ci-dessus apparaît clairement ici. Sans cette dernière, l'opération de substitution peut être confrontée à des cas d'ambiguïté, lorsque l'image d'un nœud par la fonction de correspondance n'est pas unique.

La construction de la *SSyntP* à partir d'une *SSyntS* se résume donc à remplacer dans la *SSyntS Analytique* chaque nœud de surface par le nœud profond lui correspondant par relation d'abstraction. C'est une opération très peu coûteuse à mettre en œuvre informatiquement.

5.2.3 Deux Problèmes apparaissant lors de la construction de la *SSyntP*

Le processus de construction de la *SSyntP Analytique*, tel que nous l'avons décrit, va être confronté à deux problèmes : le premier concerne la définition des règles d'abstraction ; le second se pose pour certaines configurations de modifieurs. Les deux problèmes et les solutions que nous proposons sont décrits dans les deux sections suivantes.

Les arbres élémentaires de surface étendus

La description que nous avons faite de la relation d'abstraction comprenait une hypothèse implicite, qui voulait qu'à un $\mathcal{AES}$ corresponde un $\mathcal{AEP}$. Les règles d'abstraction telles que décrites s'appliquent en effet sur un unique *AES* pour produire un unique *AEP*. Cette hypothèse se trouvait vérifiée dans les exemples que nous avons proposés. La relation d'abstraction de la figure 5.22, par exemple, s'établit naturellement entre un *AES* et un *AEP* associés à un verbe. Dans certains cas, par contre, la relation d'abstraction entre un *AES* et un *AEP* est plus difficile à établir. Une telle situation se présente pour les verbes à temps complexe représentés en syntaxe de surface par deux mots, donc deux *AES*, et par un seul *AEP* en syntaxe profonde, comme l'illustre la figure 5.23. Faut-il dans ce cas lier l'*AEP* à l'*AES* de l'auxiliaire ou à celui du participe passé ?

Ce problème de correspondance provient du fait que les *AES* et les *AEP* sont définis de façon indépendante. La structure des *AES* est le résultat de contraintes propres à l'analyse syntaxique, synthétisées sous forme des trois principes de bonne formation exposés en 4.1.6. Les *AEP* sont, eux, définis en fonction des règles de paraphrase. Or il est des cas où ces structures ne se correspondent pas naturellement : les *AEP* définis par les règles de paraphrase requièrent souvent un domaine de localité plus étendu que celui des *AES*. Il est possible de contourner le problème de correspondance en étendant le domaine de localité des *AES*, de façon à faire correspondre

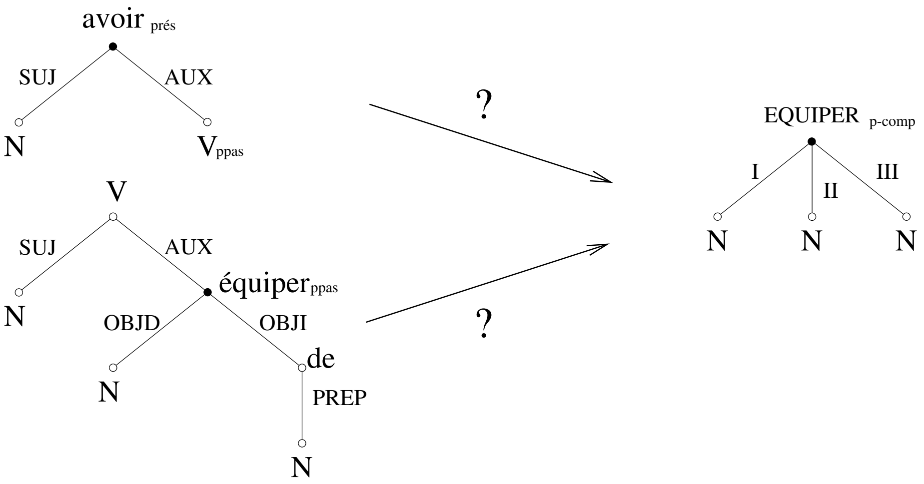


Fig. 5.23: *AES* et *AEP* ne pouvant être mis en correspondance bi-univoque

systématiquement *AES* et *AEP*. C'est une solution peu satisfaisante car les *AES* ainsi créés violeraient les principes de bonne formation des *AES* et leur structure serait difficile à justifier au regard de la tâche de l'analyse.

La solution que nous avons choisie d'adopter pour traiter ce genre de cas est de définir un troisième type d'arbre élémentaire : les *arbres élémentaires de surface étendus* (*AES+*), composés d'une combinaison de plusieurs *AES*. Les arbres élémentaires de surface étendus permettent d'établir un lien indirect entre *AES* et *AEP* : plusieurs *AES* sont regroupés au sein d'un *AES+* et ce dernier est lié à un *AEP*. Le regroupement de plusieurs *AES* au sein d'un *AES+* est établi à l'aide d'un nouveau type de règles, appelé *règle de composition* qui met en relation deux *AES* combinés d'une certaine façon et un *AES+*. Plus précisément, ces règles sont définies entre $\mathcal{AES}$ et $\mathcal{AES}+$, elles mettent en correspondance une combinaison de deux $\mathcal{AES}$ et un $\mathcal{AES}+$. La combinaison de deux $\mathcal{AES}$, $\mathcal{AES}_1$ et $\mathcal{AES}_2$, est représentée par l'expression $Unif(\mathcal{AES}_1@\alpha, \mathcal{AES}_2@\beta)$. Cette dernière indique quels nœuds des deux $\mathcal{AES}$ unifier pour former l'$\mathcal{AES}+$. La règle de combinaison comporte de plus deux fonctions partielles f_1 et f_2, chacune liant les nœuds d'un $\mathcal{AES}$ aux nœuds de l'$\mathcal{AES}+$[11] :

$$Unif(\mathcal{AES}_1@\alpha, \mathcal{AES}_2@\beta) \Rightarrow \mathcal{AES}+ \mid f_1, f_2$$

La règle permettant de composer les deux *AES* de la figure 5.23 s'exprimera de la façon suivante :

$$Unif([v.ppas]_{n_1}@0, ppas_{n_2\grave{a}n_3}@0) \Rightarrow [v.ppas]_{n_1 n_2 \grave{a} n_3} \mid f_1, f_2$$

[11]Les règles de combinaison peuvent être étendues sans difficulté pour représenter des combinaisons de plus de deux *AES*.

où $[v.ppas]_{n_1}$ est l'$\mathcal{AES}$ associé à l'auxiliaire, $ppas_{n_2 \grave{a} n_3}$, l'$\mathcal{AES}$ associé au participe passé et $[v.ppas]_{n_1 n_2 \grave{a} n_3}$, l'$\mathcal{AES}$+ associé au verbe à temps complexe. Les fonctions de correspondance f_1 et f_2 sont réduites, dans ce cas, à la fonction identité. La règle dit que lorsque deux *AES* correspondant à un auxiliaire et à un participe passé sont reliés entre eux dans une *SSyntS Analytique*, alors ils doivent être remplacés par un *AES+* associé à un verbe à temps complexe. L'application de la règle aux deux *AES* de la figure 5.23 est représentée dans la figure 5.24. Dans la partie gauche de la figure sont représentés les deux *AES* reliés entre eux au niveau de la racine (lien représenté par la flèche en pointillés) et dans la partie droite, un *AES+*. Il est important de remarquer que la différence entre *AES* et *AES+* est une différence de domaine de localité et non une différence de niveau de représentation. Il s'agit, dans les deux cas, de syntaxe de surface.

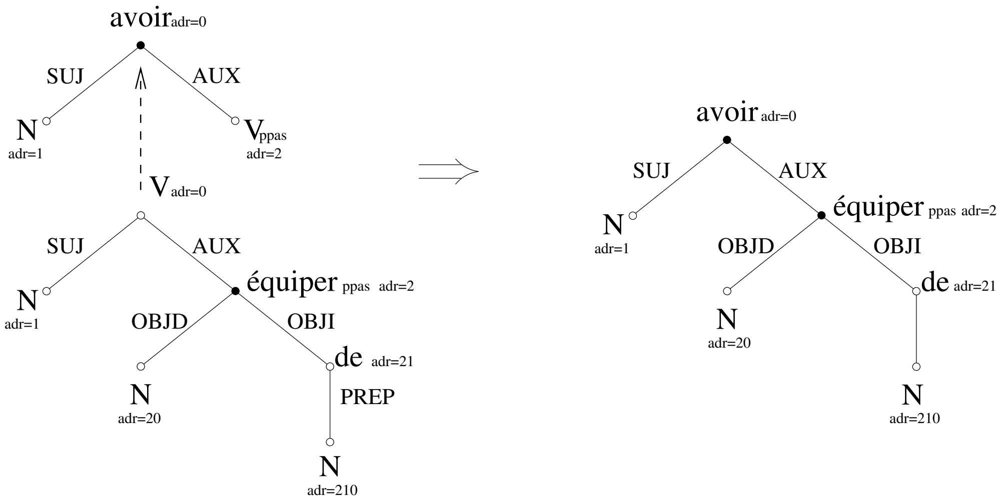


FIG. 5.24: Deux *AES* reliés à un *AES+* par relation de composition

Les règles d'abstraction qui étaient définies entre $\mathcal{AES}$ et $\mathcal{AEP}$ peuvent maintenant lier, comme avant, un $\mathcal{AES}$ à un $\mathcal{AEP}$ ou bien alors, un $\mathcal{AES}$+ à un $\mathcal{AEP}$, auquel cas elles seront représentées de la façon suivante :

$$\mathcal{AES}+ \Rightarrow \mathcal{AEP} \ \mid \ f$$

La règle d'abstraction de la figure 5.22 s'exprimera de la façon suivante :

$$[v.ppas]_{n_1 n_2 \grave{a} n_3} \Rightarrow V_{N_I N_{II} N_{III}} \ \mid \ f$$

avec

$$f(0) = 0\ ,\ f(1) = 1\ ,\ f(20) = 2\ ,\ f(210) = 3$$

L'application de cette règle d'abstraction à l'*AES+* de la figure 5.24 est illustrée dans la figure 5.25.

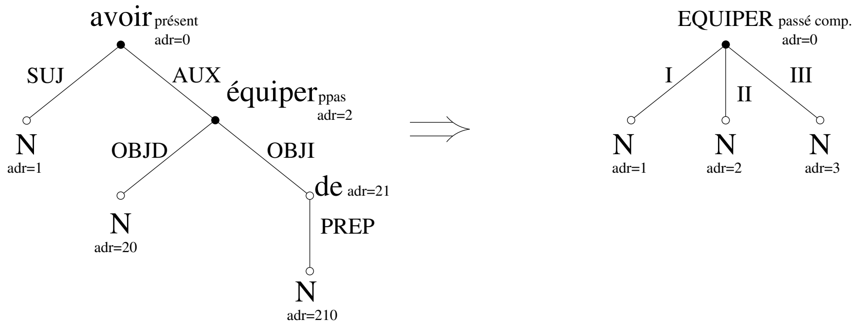


Fig. 5.25: Un *AES+* lié à un *AEP* par relation d'abstraction

L'introduction de ces nouvelles structures que sont les *AES+* et leur prise en compte dans le processus de reformulation va donc ajouter une étape à ce dernier. Après construction d'une *SSyntS Analytique*, certains *AES* seront regroupés pour constituer des *AES+*. A ce niveau du processus, la structure engendrée est hétérogène, constituée d'*AES* et d'*AES+*, car tous les *AES* ne sont pas regroupés au sein d'*AES+*. C'est alors que sont appliquées les règles d'abstraction pour construire la *SSyntP Analytique*.

Les règles de déplacement

L'application des règles d'abstraction est confrontée à un second problème structurel, concernant certains cas de modifieurs et de circonstants. Le problème apparaît notamment pour les verbes à temps complexes : dans ces cas, un modifieur ou circonstant du participe passé en syntaxe de surface va donner lieu à un modifieur ou circonstant du nœud verbal en syntaxe profonde, du fait que l'auxiliaire et le participe passé sont représentés par un seul nœud en syntaxe profonde. Or, la fonction de correspondance entre l'*AES+* du verbe à temps complexe et son *AEP* ne définit pas de lien entre le nœud du participe passé dans l'*AES+* et celui du verbe dans l'*AEP*. A l'issue de l'application de la règle d'abstraction du verbe, on ne saura donc pas où rattacher le modifieur ou le circonstant, du fait que la fonction de correspondance ne définit pas d'image au nœud du participe passé. Une telle situation est illustée dans la figure 5.26.

On pourrait résoudre un tel problème en reliant, grâce à la fonction de correspondance les nœuds correspondant à l'auxiliaire et au participe passé au nœud verbal, dans l'*AEP*. Le lien entre le participe passé (dans l'*AES*) et le verbe (dans l'*AEP*) permettraient ainsi de rattacher à l'issue de l'application des règles d'abstraction le modifieur au nœud verbal en syntaxe profonde. Une telle solution violerait cependant la condition de bijectivité que nous avions imposée à la fonction de correspondance. Or, nous avons vu que cette condition est nécessaire pour ne pas aboutir à des situations de non déterminisme et nous verrons dans le chapitre 6 qu'une telle condition est aussi nécessaire pour garantir l'inversibilité des règles d'abstraction. La solution que nous proposons consiste à introduire un nouveau type de règles, les *règles de déplacement*, pour traiter ce

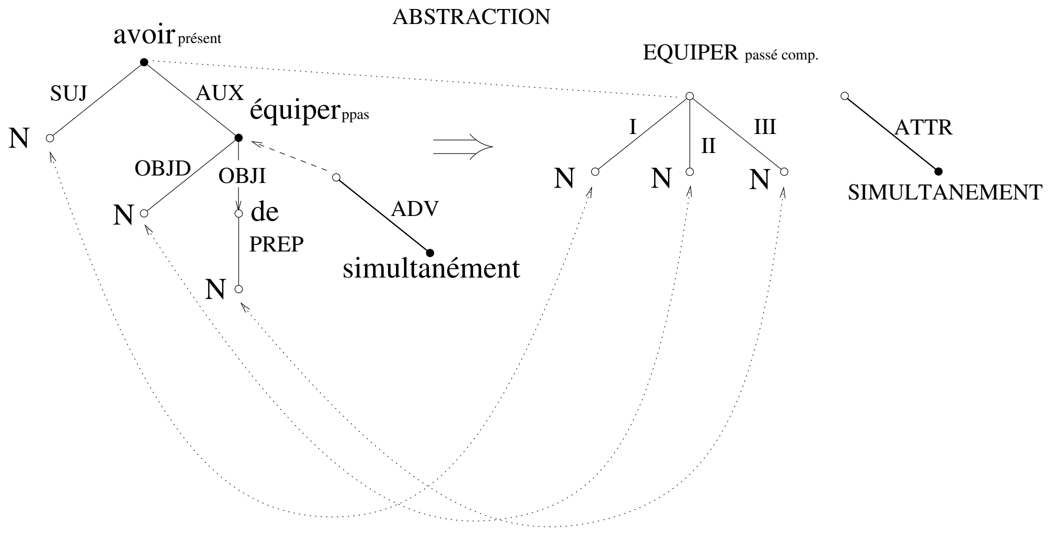


FIG. 5.26: Problème de rattachement à l'issue de l'application de règles d'abstraction

genre de phénomènes. Il s'agit de règles permettant de déplacer, dans une *SSyntS Analytique*, le site d'accrochage d'un *AES* ou *AES+* sur un autre, comme l'illustre schématiquement la figure 5.27.

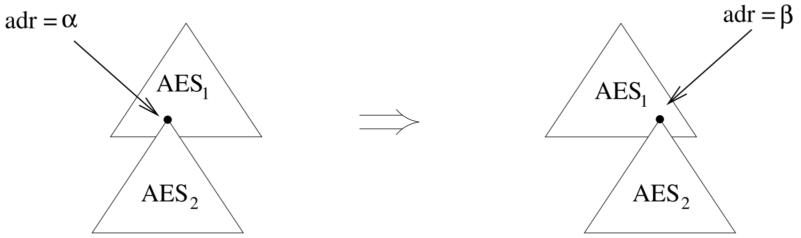


FIG. 5.27: Une règle de déplacement

La règle de la figure 5.27 est représentée de la façon suivante :

$$Unif(AES_1@\alpha\ ,\ AES_2@0) \Rightarrow Unif(AES_1@\beta\ ,\ AES_2@0)$$

Les règles de déplacement permettent de déplacer un *AES* ou *AES+* avant application des règles d'abstraction. Nous avons illustré dans la figure 5.28 l'application d'une règle de déplacement à un *AES* adverbial suivi de l'application des règles d'abstraction aux structures verbales et adverbiales. A l'issue de l'application de la règle de déplacement, l'*AES* adverbial est rattaché à l'auxiliaire, ce qui permet, lors de l'application de la règle d'abstraction, de rattacher en syntaxe profonde l'adverbe et le verbe[12].

[12]On pourrait être tenté, en observant la figure 5.28, de modifier la définition des dépendances syntaxiques

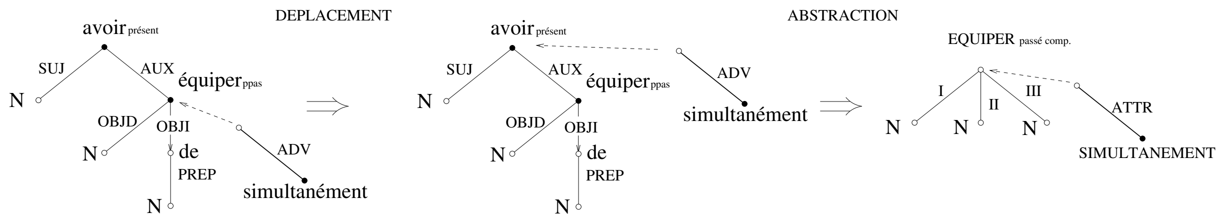


FIG. 5.28: Applications successives d'une règle de déplacement et de règles d'abstraction

Les règles de déplacement font partie d'un nouveau genre de règles, qui mettent en correspondance une combinaison de deux arbres élémentaires avec une autre combinaison des deux mêmes arbres élémentaires. Il se distingue ainsi, d'une part des règles de paraphrase et des règles d'abstraction qui s'établissent entre deux arbres élémentaires, et d'autre part des règles de composition qui permettent de regrouper deux arbres élémentaires ou plus, pour former un arbre élémentaire d'un autre type. Les règles de déplacement modifient uniquement l'agencement des *AES* au sein d'une structure analytique. Les différents types de règles introduits sont repris dans l'annexe A.

5.2.4 Les Grammaires d'arbres adjoints synchrones

On retrouve la notion de liens entre arbres élémentaires dans une variante des grammaires d'arbres adjoints (TAG), appelée TAG synchrones, proposée dans [Shieber & Schabes, 1990]. Le principe des TAG synchrones est de relier entre elles deux grammaires TAG (A et B), en reliant chaque arbre élémentaire de la grammaire A à un arbre de la grammaire B, à l'image de la figure 5.29.

Le système est dit synchrone car, lorsqu'une opération de combinaison d'arbre est effectuée dans la grammaire A, une opération équivalente est effectuée *simultanément* dans la grammaire B. La première application des TAG synchrones concernait la construction simultanée d'une représentation syntaxique et d'une représentation sémantique d'un énoncé. Le système se composait de deux grammaires : une grammaire « linguistique » et une grammaire « logique », à l'image des deux arbres élémentaires de la figure 5.29. Une application des TAG synchrones à la traduction automatique a été proposée dans [Abeillé et al., 1990] et une application à la paraphrase évoquée en [Abeillé, 1991].

Il serait envisageable, à l'instar des TAG synchrones, d'imaginer, dans notre cadre, une construc-

de surface, de sorte que l'adverbe soit rattaché dans la *SSyntS* directement à l'auxiliaire et non au participe passé. Une telle configuration permettrait de se passer des règles de déplacement : à l'issue de l'application de la règle d'abstraction, l'*AEP* correspondant se trouverait en effet attaché au nœud verbal. Cette modification serait uniquement motivée par des raisons procédurales : simplifier la transition *SSyntS*⟶*SSyntP*. Or nous nous sommes imposé comme contrainte en 3.1 de ne recourir à des transformations des représentations linguistiques proposées par la *TST* qu'en cas d'impossibilité de mettre en œuvre des traitements, ce qui n'est pas le cas ici, puisque le problème peut être résolu par l'introduction des règles de déplacement.

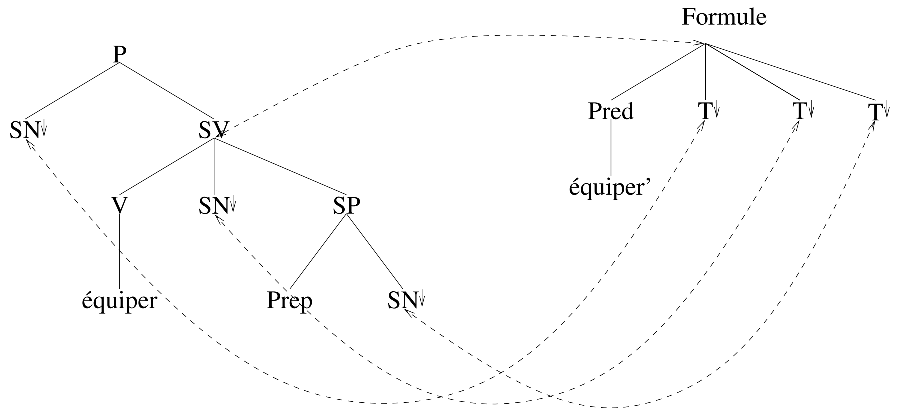


FIG. 5.29: Deux arbres élémentaires d'une grammaire TAG synchrone

tion simultanée d'une *SSyntS* et d'une *SSyntP*. Ce fonctionnement synchrone n'offre cependant aucun avantage, dans la mesure où, comme nous l'avons mentionné ci-dessus, toutes les contraintes représentées dans les *AEP* (principalement le schéma de régime) le sont aussi dans les *AES*. Ce qui veut dire qu'une opération de combinaison réussie entre deux *AES* ne peut échouer au niveau des *AEP* correspondants. La situation aurait été différente si les contraintes syntaxiques profondes n'étaient pas représentées dans les *AES*. Dans ce cas, il aurait été intéressant de construire simultanément les deux structures et l'échec de l'une aurait invalidé la seconde. L'avantage d'une telle solution étant de maintenir la distinction entre contraintes syntaxiques de surface et contraintes syntaxiques profondes tout en les prenant en compte simultanément (en accord avec les principes d'efficacité évoqués en 3.1). L'inconvénient, dans cette perspective, concerne la correspondance entre structures de surface et structures profondes. Les premières, de nature plus locale (elle pourraient être comparées aux règles syntagme d'I. Mel'čuk) seraient difficiles à mettre en correspondance avec des règles de bonne formation de la *SSyntP* (tels que les schémas de régime). On pourra à ce propos revoir l'argumentation proposée en 3.2.

Chapitre 6

Reformulation

Nous avons regroupé dans ce chapitre les deux dernières phases du processus de reformulation : la transformation de la *SSyntP* et la génération d'une ou plusieurs phrase à partir de la *SSyntP* transformée. Contrairement au chapitre précédent, qui peut être envisagé indépendamment de la problématique de la reformulation, les étapes décrites ici sont au cœur du processus de reformulation puisqu'elles modifient les structures syntaxiques de la phrase d'origine pour en proposer une paraphrase. Ces deux étapes réalisent donc la reformulation, proprement dite, d'où le titre de ce chapitre. Nous commencerons par décrire le processus de transformation de la *SSyntP*. C'est à ce niveau que sont effectués les choix majeurs de reformulation. La mise en œuvre de cette étape va faire émerger certaines inadéquations du formalisme que nous avons proposé au chapitre 3. Celles-ci seront résolues par l'introduction d'un nouveau type d'arbres élémentaires et de règles de composition dans le formalisme. Nous proposerons ensuite, en 6.2, certains cas d'applications de règles de paraphrase menant à des structures agrammaticales. Les raisons de ces problèmes seront analysées sans toutefois toujours y apporter de solutions. Dans certains cas, en effet, la solution à ces problèmes nécessite une remise en cause importante de certains aspects de la *TST*, dépassant le cadre de cette thèse. Nous décrirons en 6.3 la phase de regénération d'une phrase à partir d'une *SSyntP* transformée qui constitue la dernière partie de la reformulation. Les enrichissements du formalisme et du processus présentés dans ce chapitre, ajoutés à ceux du chapitre précédent ont considérablement complexifié le processus de reformulation. Nous évaluerons en 6.4 cet accroissement de la complexité et ses conséquences.

6.1 Transformation de la *SSyntP*

Le principe de transformation d'une *SSyntP* a été introduit au chapitre 3. Il consiste à remplacer, dans la *SSyntP Analytique*, les *AEP* interdits par les *AEP* autorisés leur correspondant par règle de paraphrase. L'exemple du chapitre 3 a été repris dans la figure 6.1. La transformation se réduit dans ce cas au remplacement de l'*AEP* associé au verbe EQUIPER par l'*AEP* correspondant au verbe INSTALLER. Ce remplacement s'accompagne de la permutation des actants II et III du verbe.

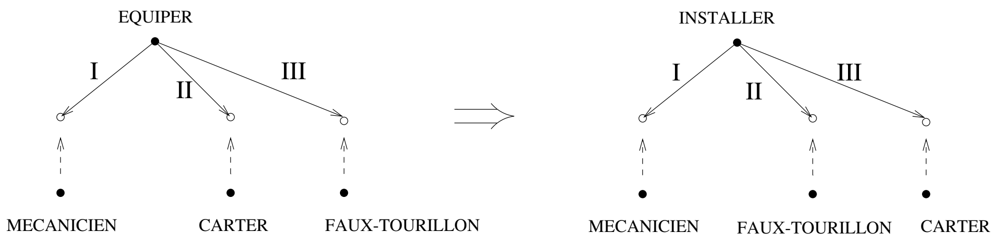


FIG. 6.1: Deux *SSyntP Analytiques*

Ce principe de transformation se rapproche du mode de construction d'une *SSyntP* à partir d'une *SSyntS* vu au chapitre 5. Dans le premier cas, il s'agit de remplacer des *AES* par des *AEP* et dans le second, des *AEP* interdits par des *AEP* autorisés. Deux différences importantes entre les deux processus méritent toutefois d'être mentionnées : contrairement à la phase de construction d'une *SSyntP* à partir d'une *SSyntS*, la phase de transformation de la *SSyntP* consiste en une transformation de structure, sans changement de niveau de représentation. De plus, alors que dans le cas de la construction de la *SSyntP*, tous les *AES* entrant dans la composition de la *SSyntS Analytique* sont remplacés par des *AEP*, dans le cas de la transformation de la *SSyntP*, seuls les *AEP* interdits sont remplacés. Il s'agit donc d'un processus sélectif qui ne modifie qu'une partie de la structure. Le caractère partiel de cette transformation permet aussi de la distinguer de la phase de transfert d'un système de traduction automatique à laquelle nous l'avons comparée jusqu'alors. Lors d'un transfert, c'est l'intégralité de la structure qui doit être modifiée alors que dans notre cas, comme nous l'avons mentionné, la transformation n'affecte qu'une partie de la *SSyntP*.

La transformation de la *SSyntP Analytique* va se décomposer en trois étapes, une étape de génération d'*AEP* à partir des *AEP* interdits grâce aux règles de paraphrase, une étape de choix d'un *AEP* parmi les différents *AEP* générés et, finalement, la substitution des *AEP*. Il est important de noter que les *AEP* interdits ne pourront pas toujours donner naissance à des *AEP* autorisés. Rien ne garantit donc qu'à l'issue de la transformation de la *SSyntP*, cette dernière soit exclusivement composée d'*AEP* autorisés.

6.1.1 Les règles de paraphrase

A l'instar des règles d'abstraction du chapitre 5 et des règles de [Gladkij & Mel'čuk, 1975], nous représenterons les règles de paraphrase comme une correspondance entre deux types d'arbres élémentaires, la correspondance étant établie ici entre deux $\mathcal{AEP}$. Les règles de paraphrase reposent, comme nous l'avons vu en 2.1.4, sur une fonction lexicale paradigmatique. Dans le cas d'une règle de remplacement d'un nom par un synonyme, par exemple, la fonction lexicale en jeu est la fonction synonyme (*Syn*). Dans la représentation des règles sous forme de correspondance entre arbres élémentaires, la condition sur l'existence d'une telle fonction lexicale s'exprime à l'aide d'une équation attachée à la règle. Cette équation met en jeu un lexème de l'*AEP* de la partie gauche de la règle et un lexème de l'*AEP* de sa partie droite, les deux lexèmes étant représentés par les nœuds qui les portent. Lorsque la condition ne peut être vérifiée, la règle échoue. Les règles de paraphrase se présentent donc de la façon suivante :

$$\mathcal{AEP}_1 \Rightarrow \mathcal{AEP}_2 \mid f$$
$$\mathcal{AEP}_1@\alpha = FL_{par}(\mathcal{AEP}_2@\beta)$$

La condition $\mathcal{AEP}_1@\alpha = FL_{par}(\mathcal{AEP}_2@\beta)$ stipule que les lexèmes portés respectivement par le nœud d'adresse α de $\mathcal{AEP}_1$ et le nœud d'adresse β de $\mathcal{AEP}_2$ doivent être liés entre eux par la fonction lexicale paradigmatique FL_{par}. Nous avons repris dans la figure 6.2 la règle de fission à verbe support présentée au chapitre 3 .

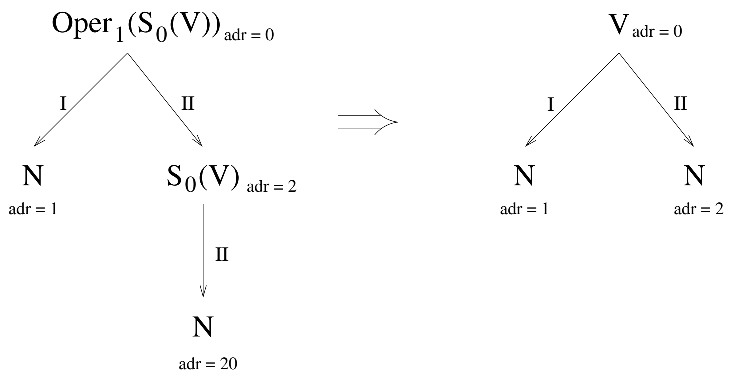


FIG. 6.2: Représentation graphique d'une règle de paraphrase

Une telle règle de paraphrase est représentée de la façon suivante :

$$[Oper_1 N]_{N_I N_{II}} \Rightarrow V_{N_I N_{II}} \mid f$$
$$[Oper_1 N]_{N_I N_{II}} @2 = S_0(V_{N_I N_{II}} @0)$$

où $[Oper_1 N]_{N_I N_{II}}$ est le type de l'*AEP* correspondant à la construction à verbe support et $V_{N_I N_{II}}$, le type de l'*AEP* correspondant à la construction verbale bivalente. La condition associée à la règle assure que la nominalisation de la structure à verbe support (portée par le nœud $[Oper_1 N]_{N_I N_{II}}$ @2) est dérivée du verbe de la structure verbale simple (porté par le nœud $V_{N_I N_{II}}$@0), grâce à la fonction lexicale S_0. La fonction de correspondance f est définie de la façon suivante :

$$f(0) = 0 \ , \ f(1) = 1 \ , \ f(20) = 2$$

L'application d'une telle règle à la construction à verbe support *effectuer une mesure* est représentée dans la figure 6.3.

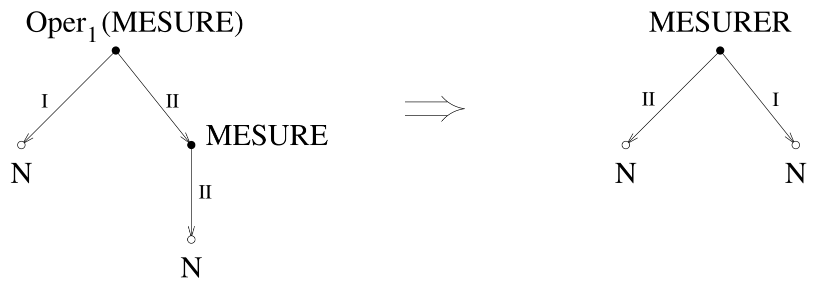


FIG. 6.3: Application d'une règle de paraphrase

Cette règle de paraphrase révèle deux problèmes :

- D'une part, elle met en jeu un $\mathcal{AEP}$ ($[Oper_1 N]_{N_I N_{II}}$) dans lequel figure un verbe support représenté par la fonction lexicale syntagmatique $Oper_1$. Or l'occurrence d'une telle fonction lexicale n'a pas été « décelée » lors de la construction de la *SSyntP*, ni représentée dans la *SSyntP*.

- D'autre part, un problème similaire à celui auquel nous avons été confronté au chapitre précédent, lors de la mise en correspondance d'un $\mathcal{AES}$ et d'un $\mathcal{AEP}$ dans le cadre des règles d'abstraction, apparaît ici. En effet, l'$\mathcal{AEP}$ de la construction à verbe support ne correspond pas structurellement à un $\mathcal{AEP}$ obtenu par abstraction d'un $\mathcal{AES}$ ou $\mathcal{AES}$+. Dans le cas de la construction *effectuer une mesure*, par exemple, l'*AEP* sur lequel s'applique la règle de paraphrase ne correspond pas à l'abstraction d'un *AES+*. Nous avons repris dans la partie droite de la figure 6.4 l'*AEP* sur lequel appliquer la règle de paraphrase et avons représenté à sa gauche l'*AES+* qui devrait lui correspondre et à la gauche de ce dernier, la combinaison d'*AES* permettant de constituer cet *AES+*. Or une telle combinaison d'*AES* n'a pas été regroupée au sein d'un *AES+* par une règle de composition lors de la phase de construction d'*AES+*, car rien ne justifiait ce regroupement dans le cadre de la transition *SSyntS* $\longrightarrow$ *SSyntP*. L'*AEP* correspondant n'a, par conséquent, pas été créé.

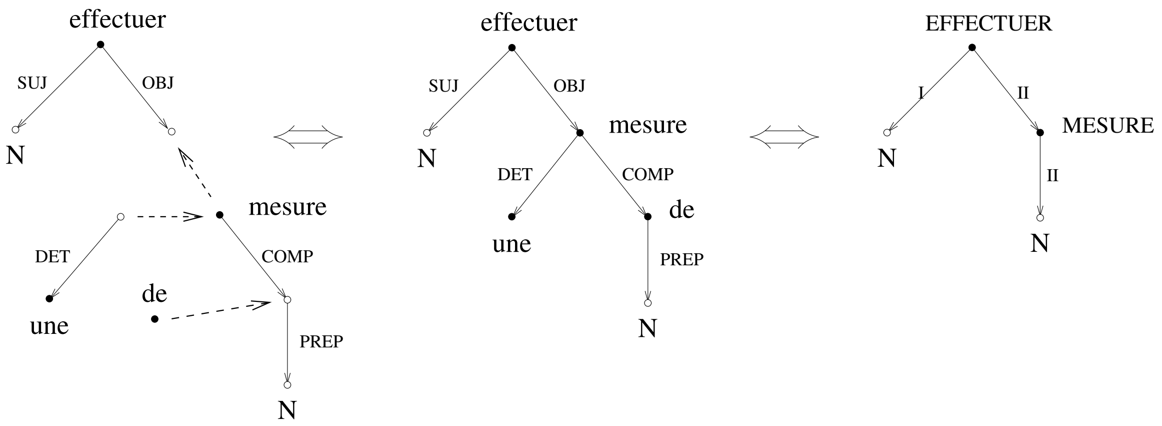


FIG. 6.4: Un *AEP* correspondant à un *AES+* mal formé

6.1.2 Les arbres élémentaires profonds étendus

Pour résoudre les deux problèmes de détection de fonctions lexicales syntagmatiques et de correspondance structurelle entre *AEP* obtenus par abstraction d'*AES* ou d'*AES+*, et d'*AEP* figurant dans les règles de paraphrase, nous allons avoir recours à une méthode similaire à celle du chapitre précédent. L'ensemble des règles entre arbres élémentaires va être enrichi d'un nouveau type de règle, les *règles de composition profonde*, permettant de combiner deux *AEP* au sein de structures plus étendues : les *arbres élémentaires profonds étendus* ou *AEP+*. A l'instar des règles de composition du chapitre précédent (que nous appellerons maintenant *règles de composition de surface*), les règles de composition profonde présentent dans leur partie gauche une composition de deux $\mathcal{AEP}$ et dans leur partie droite un $\mathcal{AEP}+$. Elles comprennent aussi deux fonctions de correspondance f_1 et f_2 permettant d'établir les liens nécessaires entre les nœuds des arbres élémentaires mis en jeu[1]. De plus, une condition associée à la règle impose l'existence d'une fonction lexicale syntagmatique liant deux lexèmes des $\mathcal{AEP}$ de la règle. Une règle de composition profonde est représentée de la façon suivante :

$$\begin{gathered} Unif(\mathcal{AEP}_1@\delta\ ,\ \mathcal{AEP}_2@\alpha) \Rightarrow \mathcal{AEP}+\ \mid\ f_1\ ,\ f_2 \\ \mathcal{AEP}_1@\beta = FL_{syn}(\mathcal{AEP}_2@\gamma) \end{gathered}$$

Une telle règle stipule que deux *AEP* de type $\mathcal{AEP}_1$ et $\mathcal{AEP}_2$ reliés entre eux dans la *SSyntP Analytique* (relation représentée dans la règle par $Unif(\mathcal{AEP}_1@\delta\ ,\ \mathcal{AEP}_2@\alpha)$) doivent être combinés au sein d'un *AEP+* de type $\mathcal{AEP}+$ si les lexèmes portés par les deux nœuds $\mathcal{AEP}_1@\beta$ et $\mathcal{AEP}_2@\gamma$ sont liés entre eux par la fonction lexicale syntagmatique FL_{syn}. Les règles de composition profonde jouent donc deux rôles : un rôle structurel permettant de regrouper deux *AEP* au sein d'un *AEP+* et un rôle de détection d'occurrence de fonctions lexicales syntagmatiques. La détection des fonctions lexicales syntagmatiques est donc réalisée durant la phase de composition d'*AEP* au sein d'*AEP+* et non durant la phase de construction de la *SSyntP*, comme c'est le cas

[1] A l'instar des règles de composition de surface, les règles de composition profonde peuvent être étendues sans difficulté au regroupement de plus de deux arbres élémentaires.

dans la *TST*.

La règle permettant de combiner deux $\mathcal{AEP}$ pour obtenir l'$\mathcal{AEP}$+ de la figure 6.4 s'exprime de la façon suivante :

$$Unif(V_{N_I N_{II}}@2\ ,\ N_{N_{II}}@0) \Rightarrow [Oper_1 N]_{N_I N_{II}}\ \mid\ f_1\ ,\ f_2$$
$$V_{N_1 N_{II}}@0 = Oper_1(N_{N_{II}}@0)$$

où $V_{N_I N_{II}}$ est le type d'*AEP* associé à un verbe bivalent (le verbe support), $N_{N_{II}}$ est le type d'*AEP* associé à un nom monovalent (la nominalisation) et $Oper_1$ est la fonction lexicale syntagmatique liant une nominalisation à un de ses verbes support. Cette règle est représentée graphiquement dans la figure 6.5. Les fonctions de correspondance f_1 et f_2 sont définies de la façon suivante :

$$f_1(0) = 0\ ,\ f_1(1) = 1$$
$$f_2(0) = 2\ ,\ f_2(1) = 20$$

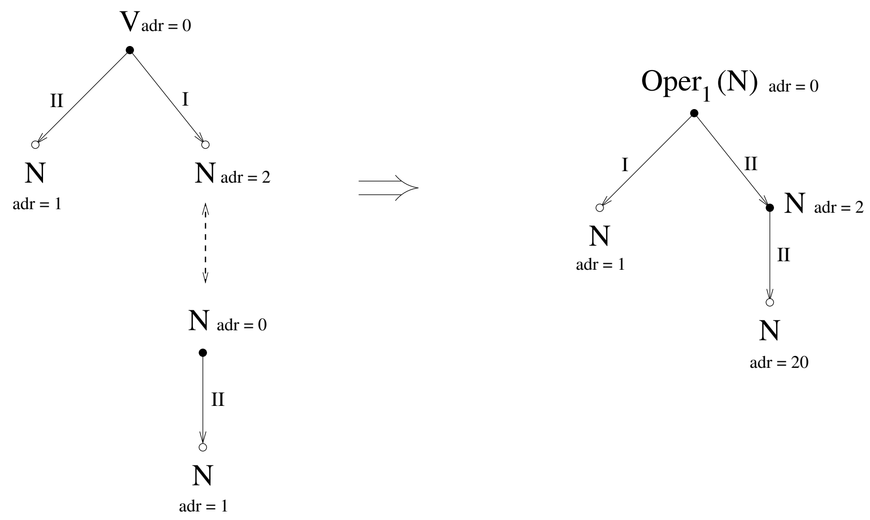


FIG. 6.5: Une règle de composition profonde

Nous avons représenté dans la figure 6.6 l'application de cette règle à la composition de deux *AEP*, un *AEP* associé à un verbe (EFFECTUER) et un *AEP* associé à un nom (MESURE). A l'issue de l'application de la règle de composition, les deux *AEP* sont regroupés au sein d'un *AEP+* et EFFECTUER est reconnu comme verbe support de MESURE.

Il aurait été possible de faire l'économie des règles de composition profonde et de traiter les regroupements effectués par ces dernières au niveau syntaxique de surface, grâce aux règles de composition de surface. Nous aurions pu, en effet, regrouper dans la *SSyntS Analytique*, les quatre *AES* de la figure 6.4 correspondant à la construction à verbe support au sein d'un *AES+*, de la même manière que nous avions regroupé au sein d'un *AES+* les *AES* correspondant respectivement à un auxiliaire et à un participe passé. Cette solution ne nous a pas semblé souhaitable à deux titres. D'une part, elle implique de traiter par une même règle des phénomènes distincts que sont la détection de phénomènes de cooccurrence lexicale restreinte et le regroupement de plusieurs lexèmes de surface au sein d'un lexème profond, tel que le regroupement d'un auxiliaire et d'un participe passé. D'autre part, cette solution aurait provoqué une multiplication du nombre de règles de composition de surface et masqué un niveau de factorisation. En effectuant les regroupements au niveau syntaxique de surface, il aurait été nécessaire de prévoir, par exemple, une règle de détection des structures à verbe support pour les temps simples et une autre pour les temps complexes alors que dans notre cas, une seule règle suffit pour les temps simples et les temps complexes.

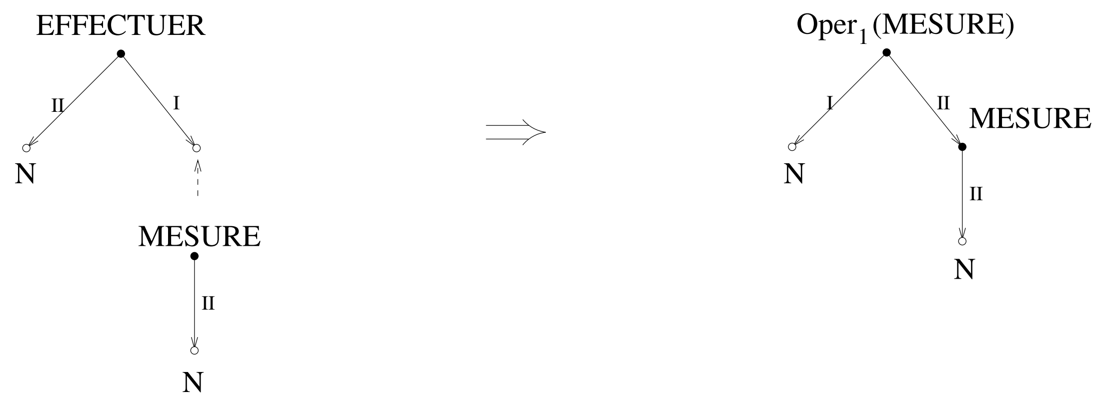


FIG. 6.6: Application d'une règle de composition profonde

Les règles de paraphrase qui étaient définies entre *AEP* peuvent maintenant être définies, selon leurs exigences, entre *AEP* ou *AEP+*. Une règle de paraphrase conversive (permettant de remplacer un verbe par un converse) s'établit entre deux *AEP* alors qu'une règle de fusion à verbe support (remplacement d'une construction à verbe support par un verbe simple) s'établit entre entre un *AEP+* et un *AEP*.

L'application de règles de paraphrase à une *SSyntP Analytique* se décompose donc en deux étapes : une étape de regroupement d'*AEP* au sein d'*AEP+* grâce aux règles de composition profondes et une phase d'application des règles de paraphrases aux *AEP+* ainsi créés. Rappellons qu'à l'issue de l'application des règles de composition profonde, seuls certains *AEP* se trouvent regroupés au sein d'*AEP+*.

6.1.3 Les règles d'ajustement

L'application de certaines règles de paraphrase peut entraîner la construction de structures incohérentes. Nous avions donné en 2.2.3 un exemple de fusion à verbe support où une construction à verbe support était remplacée par une construction comportant un verbe sémantiquement plein (*Le mécanicien effectue une mesure précise de tension* $\longrightarrow$ **Le mécanicien mesure précise la tension*). Pour générer une phrase correcte, une telle règle doit prévoir la transformation de l'adjectif *précise* en un adverbe (*précisément*) ou en un syntagme prépositionnel (*avec précision*) modifiant le verbe de la nouvelle phrase. De telles transformations ne sont actuellement pas effectuées dans le système de paraphrase de la *TST*.

Elles sont réalisées dans notre cadre par un nouveau genre de règle. Il ne s'agit pas à proprement parler de règles de paraphrase dans la mesure où elles ne permettent pas la transformation d'une structure en une structure paraphrastique, mais plutôt de *règles d'ajustement* qui accompagnent les règles de paraphrase et dont l'application est subordonnée à celle de ces dernières. Les règles d'ajustement garantissent en particulier la bonne forme des modificateurs et permettent le bloquage de certaines paraphrases. Les règles d'ajustement recouvrent deux aspects : une transformation d'*AEP* ou d'*AEP+*, telle que la transformation d'un adjectif en un adverbe ou en un syntagme prépositionnel et le déplacement de la structure transformée. Ces deux aspects seront traités par deux types de règles, regroupés au sein d'une règle d'ajustement : des *règles de dérivation* prenant en charge les transformations d'*AEP* et des *règles de déplacement* qui ont déjà été introduites au chapitre 5. Une règle d'ajustement est composée d'une règle de dérivation et d'une règle de déplacement. Nous avons représenté schématiquement dans la figure 6.7 les effets d'une règle d'ajustement. L'*AEP* associé à l'adjectif PRÉCIS est transformé en un *AEP* associé à l'adverbe PRÉCISÉMENT. De plus, le site d'attachement de la structure dérivée est modifié.

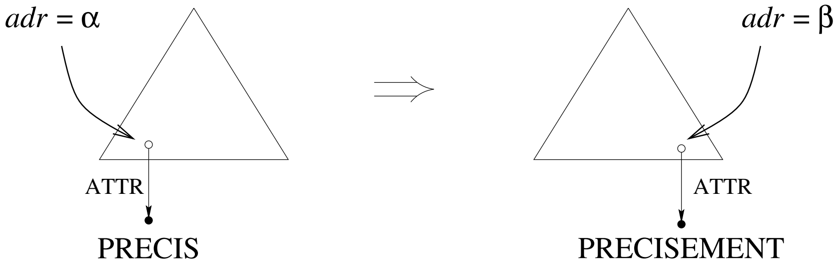


FIG. 6.7: Une règle d'ajustement

Les règles de dérivation

Les règles de dérivation sont représentées par une correspondance entre $\mathcal{AEP}$ ou $\mathcal{AEP}$+ enrichie d'une fonction de correspondance et d'une condition sur l'existence d'une fonction lexicale paradigmatique (telle que la fonction lexicale reliant un nom et son adjectif dérivé) liant un nœud de l'arbre de la partie droite de la règle à un nœud de l'arbre de sa partie gauche. Une règle établie entre deux $\mathcal{AEP}$ s'écrit de la façon suivante :

$$\mathcal{AEP}_1 \Rightarrow\ \mathcal{AEP}_2 \mid f$$
$$\mathcal{AEP}_1 @\alpha = FL_{par}(\mathcal{AEP}_2 @\beta)$$

Le succès de l'application de la règle à un *AEP* ancré par un lexème profond L particulier est alors subordonné à l'existence de la fonction lexicale pour le lexème L. La règle de dérivation permettant de dériver d'une structure adjectivale une structure prépositionnelle en *avec*, que l'on appellera $\mathcal{R}déri v_1$, s'écrit de la façon suivante :

$$\mathcal{R}dériv_1 \;:\; ADJ_{ATTR} \Rightarrow \; avec_N \mid f$$
$$ADJ_{ATTR}@\beta = Adj_0(avec_N@\alpha)$$

ADJ_{ATTR} étant l'$\mathcal{AEP}$ associé à un adjectif en position d'épithète, $avec_N$, l'$\mathcal{AEP}$ associé à un syntagme prépositionnel en introduit par la préposition *avec* et Adj_0, une fonction lexicale paradigmatique reliant un nom et son adjectif dérivé. L'application de la règle à l'*AEP* de l'adjectif *précis* est représentée dans la figure 6.8. Dans ce cas, l'application de la règle réussit car Adj_0(*précision*) existe dans le lexique : sa valeur est *précis*[2].

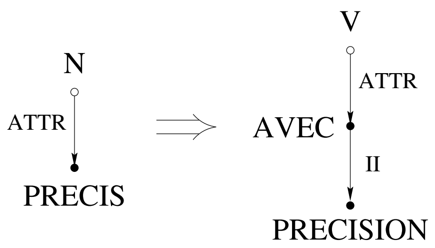


FIG. 6.8: Application d'une règle de dérivation

Les règles d'ajustement se présentent comme une combinaison d'une règle de déplacement et d'une règle de dérivation :

$$Unif(\mathcal{AEP}_1@\alpha \;,\; \mathcal{AEP}_2@0) \Rightarrow Unif(\mathcal{AEP}_1@\beta \;,\; \mathcal{AEP}_3@0)$$
$$\mathcal{AEP}_3 = \mathcal{R}dériv(\mathcal{AEP}_2)$$

La règle assure le déplacement de $\mathcal{AEP}_2$ d'un site d'adresse α vers un site d'adresse β ainsi que sa transformation en $\mathcal{AEP}{+}_3$ par application de la règle de dérivation $\mathcal{R}dériv$. Cette dernière joue aussi un rôle de condition : lorsque son application échoue, la règle d'ajustement échoue aussi.

La transformation de l'épithète d'une nominalisation dans une structure à verbe support sera traitée par la règle d'ajustement $\mathcal{R}ajust_1$ suivante :

$$\mathcal{R}ajust_1 \;:\; Unif([Oper_1N]_{N_IN_{II}}@2 \;,\; ADJ_{ATTR}@0) \Rightarrow Unif([Oper_1N]_{N_IN_{II}}@0 \;,\; X@0)$$
$$X = \mathcal{R}dériv_1(ADJ_{ATTR})$$

Elle stipule que lorsqu'un adjectif est rattaché à la nominalisation d'un verbe support ($Unif([Oper_1N]_{N_IN_{II}}@2 \;,\; ADJ_{ATTR}@0)$), il doit être remplacé par un syntagme prépositionnel ($\mathcal{R}dériv_1(ADJ_{ATTR})$) et rattaché au verbe support de la construction ($Unif([Oper_1N]_{N_IN_{II}}@0 \;,\; X@0)$).

[2] Les règles de dérivation peuvent être rapprochées de la translation chez [Tesnière, 1959].

6.1.4 Les règles généralisées de paraphrase

Les règles de paraphrase enrichies de règles d'ajustement constituent maintenant des entités complexes, que nous appellerons *règles généralisées de paraphrase.* Ces dernières sont constituées d'une règle de paraphrase et d'éventuelles règles d'ajustement. Les règles d'ajustement jouent aussi un rôle de condition d'application de la règle de paraphrase. Nous les représenterons de la façon suivante :

Règle généralisée de paraphrase:

Règle de paraphrase
{Règle d'ajustement1}
{Règle d'ajustement2}
...

La règle généralisée de fission/fusion à verbe support s'écrira de la façon suivante :

$$[Oper_1 N]_{N_I N_{II}} \Rightarrow V_{N_I N_{II}} \;\; | \;\; f$$
$$[Oper_1 N]_{N_I N_{II}} @2 = S_0(V_{N_I N_{II}} @0)$$
$$\{\mathcal{R}ajust_1\}$$

où $\mathcal{R}ajust_1$ est la règle d'ajustement présentée ci-dessus.

Son application à l'exemple *Le mécanicien effectue une mesure précise de tension* est représentée dans la figure 6.9.

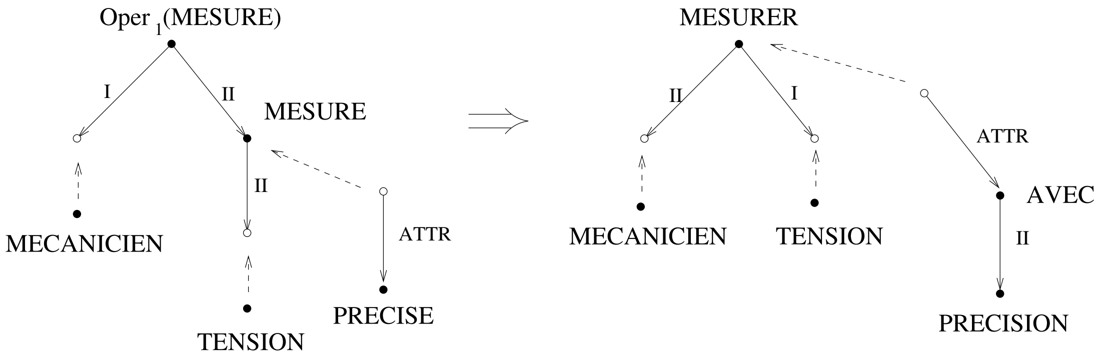


FIG. 6.9: Application d'une règle généralisée de paraphrase

Les règles généralisées de paraphrase permettent de faire l'économie de la seconde méta-règle du système de paraphrase présentée en 2.1.4. Cette dernière permettait de répartir, lors d'une opération de fission d'un nœud A en deux nœuds B et C, les dépendants de A entre B et C. Cette répartition est maintenant assurée par les règles d'ajustement qui permettent de plus d'effectuer les transformations nécessaires, telles que la transformation d'un adjectif en un adverbe ou en un syntagme prépositionnel.

Les règles de paraphrase généralisées ainsi que les règles d'ajustement constituent un nouveau type de règles, que nous appellerons *règles complexes* car elles se composent de règles plus simples, appelées *règles élémentaires*, que sont les règles d'abstraction, de paraphrase, de composition de surface et de déplacement. Les différents types de règles ont été repris en annexe A.

6.1.5 Inversibilité des règles de paraphrase

Les règles de paraphrase de la *TST* sont, en théorie, bidirectionnelles. Une règle permettant de paraphraser une construction verbale en une structure à verbe support, par exemple, peut être envisagée dans les deux sens. Bien que les règles telles que nous les avons définies soient unidirectionnelles, leur inversion est simple : elle consiste à permuter les parties gauche et droite de la règle et à inverser la fonction f. La condition sur la fonction lexicale paradigmatique reste, elle, inchangée. La règle de paraphrase suivante :

$$\mathcal{AEP}+_1 \Rightarrow \mathcal{AEP}+_2 \mid f$$
$$\mathcal{AEP}+_1@\alpha = FL_{par}(\mathcal{AEP}+_2@\beta)$$

peut être inversée pour donner la règle :

$$\mathcal{AEP}+_2 \Rightarrow \mathcal{AEP}+_1 \mid f^{-1}$$
$$\mathcal{AEP}+_1@\alpha = FL_{par}(\mathcal{AEP}+_2@\beta)$$

La fonction de correspondance étant bijective, son inversion est toujours possible.

L'inversion des règles généralisées de paraphrase consiste en une inversion de la règle de paraphrase qu'elle contient et des règles d'ajustement qui lui sont associées. L'inversion de ces dernières se fait aussi par permutation des membres gauches et droites et inversion de la fonction de correspondance.

6.1.6 Transformation d'une *SSyntP* complète

Nous ne nous sommes intéressés jusque là qu'à la représentation des règles de paraphrase et leur application à une structure élémentaire. Nous allons envisager ici l'application de plusieurs règles à une *SSyntP Analytique* complète. Ce processus se base sur une hypothèse concernant la localité des règles de paraphrase. On considère que l'effet de l'application d'une règle de paraphrase généralisée se borne à remplacer les *AEP* ou *AEP+* mentionnés dans la règle, sans perturber le reste de la *SSyntP*. Cette hypothèse est énoncée dans [Mel'čuk, 1988b] sous la forme d'une *méta-règle* qui stipule que« *Lors de l'application d'une règle syntaxique de paraphrase R à une structure syntaxique profonde, tout nœud de cette structure qui n'est pas mentionné dans R reste attaché à son gouverneur par la même relation syntaxique de surface, sauf mention expresse du contraire dans une autre méta-règle* ». Nous maintiendrons la même hypothèse dans notre modèle, au niveau des règles généralisées. Nous considèrerons que l'application d'une règle généralisée à une structure syntaxique profonde ne va affecter que les nœuds de cette structure qui sont mentionnés dans la règle. Cette hypothèse n'est pas vraiment équivalente à la méta-règle enoncée ci-dessus car les règles généralisées sont en général plus étendues que les règles de paraphrase de la *TST* : elles mettent en jeu plus de nœuds que les règles de paraphrase simples leur correspondant. Le processus de transformation d'une *SSyntP Analytique* se décompose en trois étapes :

1. Génération d'*AEP* ou *AEP+* autorisés

   Pour chaque *AEP* ou *AEP+* interdit (que nous appellerons A) entrant dans la composition de la *SSyntP Analytique*, la liste des règles de paraphrases pouvant lui être appliquée est établie. Ces dernières sont les règles dont la partie gauche ou droite est constituée du type de A. L'application de ces règles de paraphrase donne naissance à de nouveaux *AEP* ou *AEP+*. Parmi ces derniers, certains sont autorisés et d'autres pas. L'ensemble d'arbres élémentaires ainsi créé est trié et ne sont gardés que les arbres autorisés. Rien ne garantit, à ce niveau, l'existence d'un *AEP* ou *AEP+* autorisé. Dans un tel cas, lorsque tous les *AEP* ou *AEP+* générés sont interdits, ces derniers serviront de base pour en générer une nouvelle série, des *AEP* ou *AEP+* de deuxième génération, comme l'illustre la figure 6.10.

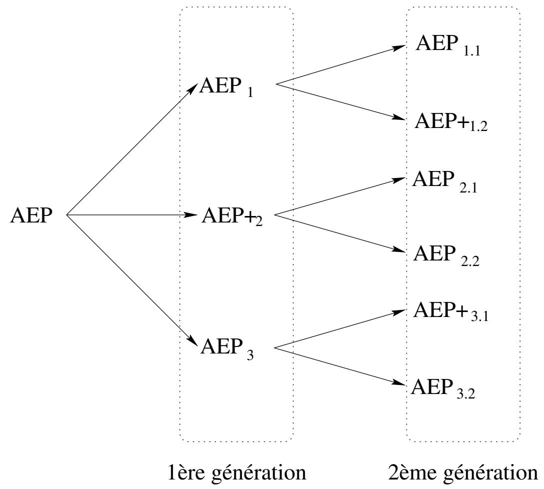


FIG. 6.10: *AEP* de première et seconde génération

   L'interêt d'un tel mécanisme peut être illustré par un exemple du français rationnalisé qui interdit les constructions à verbe support ainsi que l'emploi du verbe *installer* auquel on préfère le« synonyme» *équiper*. La construction *effectuer une installation* est donc interdite en tant que construction à verbe support. Une telle constuction devrait être remplacée par le verbe *équiper* auquel elle ne peut par contre pas être liée directement. Il n'existe en effet pas de règle de paraphrase permettant de relier ces deux constructions. Le système de génération d'*AEP+* de seconde génération, décrit ci-dessus, permet de résoudre le problème en dérivant de la construction *effectuer une installation* le verbe *installer* grâce à une règle de fusion à verbe support. Bien qu'interdite, la nouvelle structure pourra donner naissance à une construction autorisée, le verbe *équiper*, grâce à une règle de remplacement d'un verbe par un verbe converse. La situation est illustrée graphiquement dans la figure 6.11.

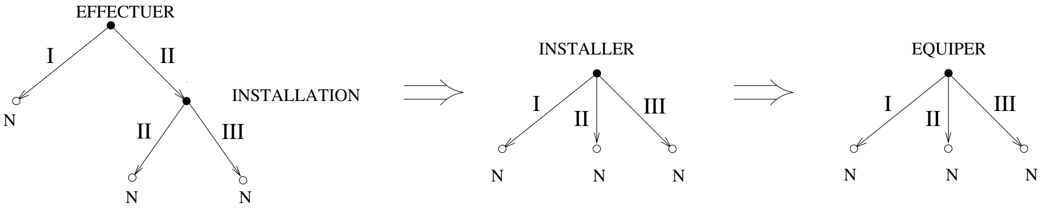


FIG. 6.11: Exemple d'un *AEP* de seconde génération

L'utilisation d'*AEP* ou *AEP+* de seconde génération va par contre accroître les risques de glissements sémantiques qui ont été évoqués en 2.2.3.

2. Choix d'un *AEP* ou *AEP+*

   Parmi les arbres élémentaires autorisés générés à l'étape précédente, un seul devra être choisi pour remplacer l'*AEP* ou *AEP+* interdit. Une première solution consiste à choisir, arbitrairement, un élément de l'ensemble établi à l'étape précédente. Nous avons préféré enrichir le marquage des *AEP* et *AEP+* en substituant à l'opposition interdit/autorisé, la notion de niveau d'interdiction. Un arbre élémentaire profond possède ainsi un niveau d'interdiction qui est d'autant plus élevé que l'usage de l'*AEP+* est déconseillé. Le processus de reformulation revient mainenant à remplacer les *AEP* ou *AEP+* par d'autres, ayant un niveau d'interdiction inférieur.

3. Remplacement des *AEP* ou *AEP+*.

   La dernière phase du processus consiste à remplacer les arbres élémentaires profonds entrant dans la composition de l'arbre de dérivation de surface par l'*AEP* ou *AEP+* qui aura été choisi à l'étape précédente. Le remplacement se fait grâce à l'opération de substitution.

6.2 Imperfections du système de paraphrase de la *TST*

Nous avions identifié en 2.2.3 deux problèmes auxquels était confrontée l'application de règles de paraphrase. Le premier concernait le manque de précision sémantique des fonctions lexicales. Le second portait sur l'absence de conditions d'application des règles de paraphrase. Ce dernier problème a été partiellement résolu par l'introduction des règles d'ajustement et des règles généralisées de paraphrase. Tous les problèmes ne sont pas pour autant résolus. Nous allons présenter dans les trois points suivants des cas d'erreurs provoquées par l'application de règles de paraphrase.

- Passif

  L'application de certaines règles de paraphrase à des verbes à la diathèse passive peut provoquer des incohérences. Le problème apparaît en particulier lors de l'application d'une règle de paraphrase conversive (permettant de remplacer un verbe par un verbe converse) à un verbe au passif. Considérons tout d'abord le remplacement du verbe *équiper* par son converse *installer* dans la phrase active suivante[3] :

  Le mécanicien équipe le carter de faux-tourillons.
  I II III

  qui aboutit à la phrase :

  Le mécanicien installe les faux-tourillons sur le carter.
  I II III

  L'effet de la règle de paraphrase a consisté à remplacer le verbe *équiper* par le verbe *installer* et à permuter les actants syntaxiques profonds II (*carter*) et III (*faux-tourillons*). La même règle appliquée à la même phrase à la diathèse passive :

  Le carter est équipé de faux-tourillons par le mécanicien.
  I III II

  engendrera la phrase

  **Le carter est installé sur le mécanicien par les faux-tourillons.*
  I II III

  Le problème provient du fait que la règle s'est bornée, ici aussi, à permuter les deux actants profonds II et III alors que les actants qui auraient dû être permutés sont les actants I (*carter*) et III (*faux-tourillons*) car le changement de diathèse a modifié l'étiquetage des dépendants du verbe. Les dépendances syntaxiques profondes n'ont en effet pas de connotation sémantique : l'actant I à la diathèse active n'ocupe pas le même rôle thématique vis-à-vis du prédicat qu'à la diathèse passive. Le problème rencontré lors de l'application de la règle de paraphrase peut être résolu assez simplement : il suffit de créer une version « passive » de la règle de paraphrase conversive. Dans sa version passive, cette dernière permuterait les actants I et III et non plus les actants II et III. Chaque règle serait de plus enrichie d'une condition sur la diathèse du verbe.

[3]Nous avons fait figurer sous les actants du verbe l'étiquette de la dépendance les reliant à ce dernier.

- Verbes admettant plusieurs schémas de régime

  Un problème proche se pose lors de l'application d'une règle de paraphrase conversive à une phrase dont le verbe principal admet plusieurs schémas de régime, tel que le lexème *payer*IIb qui admet deux schémas de régime, appelés MOD1 et MOD2, illustrés dans les deux phrases suivantes[4] :

  MOD2 : *Ce cadeau paiera Pierre pour la faveur qu'il nous a rendue*
  I II III

  MOD1 : *Ce cadeau paiera à Pierre la faveur qu'il nous a rendue*
  I III II

  Selon le schéma de régime choisi, un même actant sémantique occupe des rôles syntaxiques profonds différents. Le nom propre *Pierre*, par exemple, joue le rôle d'actant syntaxique profond II dans la première phrase et le rôle III dans la seconde, alors qu'il occupe dans les deux phrases le même rôle thématique. Un problème apparaît lors du remplacement du lexème *payer*IIb par son synonyme *récompenser*, qui aboutit à un résultat correct dans la première phrase :

  Ce cadeau récompensera Pierre pour la faveur qu'il nous a rendue.

  mais pas dans la seconde :

  **Ce cadeau récompensera à Pierre la faveur qu'il nous a rendue.*

  Le problème provient du fait que le schéma de régime du verbe *récompenser* est compatible avec un seul des schémas de régime du verbe *payer*IIb (MOD2). La solution la plus simple pour résoudre ce type de problème consiste ici aussi à dupliquer les règles de paraphrase, de sorte que chaque schéma de régime soit pris en compte par une règle. Cette solution nécessite de plus d'ajouter une condition aux règles pour limiter leur application aux schémas de régime pour lesquelles elles sont conçues, ce qui suppose, par conséquent, une convention de nommage des schémas de régime. MOD1, par exemple, représentera un type de schéma de régime particulier.

  [Béringer, 1988] cite un problème similaire que pose l'application de règles de fusion/fission à verbe support sur des verbes admettant plusieurs schémas de régime. Là aussi, l'application de la règle n'aboutit à un résultat correct pour un seul schéma de régime.

- Constructions à verbes supports

  Les règles d'ajustement et de paraphrase généralisées ont été introduites, entre autres, pour résoudre des problèmes liés aux modifieurs dans des structures à verbes supports. C'est une règle généralisée qui permet, par exemple, de paraphraser la phrase :

  Le mécanicien effectue une mesure précise de tension.

[4] Ces deux exemples ont été tirés de l'article de [Mel'čuk et al., 1987] consacré au lexème *payer*IIb.

en

Le mécanicien mesure la tension avec précision.

Une telle règle ne peut toutefois être appliquée systématiquement. Ainsi dans l'exemple

Le mécanicien effectue un test dimensionnel de la pièce.

qui présente la même structure sytaxique que l'exemple précédent, l'application de la règle généralisée de fusion à verbe support décrite ci-dessus aboutira aux phrases incorrectes suivantes :

**Le mécanicien teste dimensionnellement la pièce.*

si on considère que l'adverbe *dimensionnellement* existe ou

** Le mécanicien teste avec dimension la pièce.*

alors que la paraphrase correcte pourrait être

Le mécanicien teste les dimensions de la pièce.

L'appplication de la règle ci-dessus doit donc être bloquée afin de prévenir la construction d'une structure incorrecte. Dans notre cas, la condition doit porter sur la nature sémantique de la relation qu'entretiennent la nominalisation et son épithète (dans un cas *mesure* et *précise* et dans l'autre *test* et *fonctionnel*). Dans le premier cas, il s'agit d'une relation de manière mais pas dans le second. Cette différence n'est pas visible au niveau de la *SSyntP* : les noms sont liés à leurs épithètes, dans les deux cas, par la dépendance ATTR et la *TST* ne propose pas de système de traits associés aux lexèmes permettant d'opposer sémantiquement ces deux adjectifs. Cette absence d'informations d'ordre sémantique au niveau de la *SSyntP* rend malaisée l'écriture de ce type de conditions d'application des règles de paraphrase[5].

La difficulté de manipuler des constructions à verbe support a été étudié en détail par [Danlos, 1992] dans le cadre du projet de traduction automatique EUROTRA ([EUROTRA, 1991]). Certains noms prédicatifs français, tels que *croc-en-jambe* n'ont en effet pas d'équivalent anglais. La traduction d'une structure à verbe support ayant pour nom prédicatif *croc-en-jambe* tel que *faire un croc en jambe* vers l'anglais nécessite l'usage d'un verbe sémantiquement plein (*to trip up*). La règle de transfert nécessaire à la réalisation de la traduction peut être rapprochée d'une règle de reformulation de fusion à verbe support et les problèmes que nous avons évoqué pour la reformulation se retrouvent lors de la traduction. Ainsi, les deux phrases françaises *Jean à fait un croc-en-jambe à Marie qui était déplacé* et *Jean à fait un croc-en-jambe à Marie qui est parti du pied droit* syntaxiquement proches (modification du nom prédicatif par une relative) donneront lieu à des structures différentes en anglais (*John tripped up Mary, which was out of place* et *John tripped up Mary with his right foot*). La difficulté d'élaborer des stratégies de traductions non *ad-hoc* pour de tels cas a poussé les membres du projet EUROTRA à ne recourir à de telles règles de transfert dans les

[5]Il serait possible de différencier dans la *TST* ces deux adjectifs, grâce aux différences que présentent les réseaux sémantiques qui leurs sont associés dans le *DEC*. Ces différences sont toutefois difficiles à exploiter : elles supposent une analyse différentielle de réseaux sémantiques des deux adjectifs sur laquelle rien n'est dit dans la *TST*.

seuls cas où le nom prédicatif n'est ni modifié ni pronominalisé. Cet exemple illustre, s'il en est encore besoin, les points communs aux deux processus de reformulation et de traduction. Une étude détaillée de ces points communs dépasse malheureusement le cadre de cette thèse.

Adéquation du niveau syntaxique profond pour la paraphrase

Les exemples mentionnés ci-dessus soulèvent le problème de l'adéquation du niveau syntaxique profond pour la paraphrase. En effet, comme nous l'avons vu dans le troisième point, certaines conditions sémantiques, nécessaires au bon fonctionnement du système de paraphrase, sont impossibles à exprimer dans la *TST*. De plus, les solutions que nous avons ébauchées pour les deux premiers points ne sont guère satisfaisantes. Le dédoublement des règles de paraphrase pour prendre en compte des situations syntaxiquement proches et identiques au plan sémantique est difficile à justifier à un niveau de représentation dont l'existence se justifie, en partie, par rapport à la paraphrase. Un tel niveau de représentation devrait permettre une expression simple et économique des règles de paraphrase.

Ces problèmes militent en faveur d'un enrichissement du niveau syntaxique profond par des informations sémantiques afin de contrôler leur application. Un premier enrichissement consiste à introduire des traits sémantiques au niveau des entrées lexicales. De tels traits ont déjà été introduits dans le but de traiter certains cas d'ambiguïté lors de l'analyse. La notion de trait sémantique n'est d'ailleurs pas étrangère à la *TST* où ils sont introduits notamment pour imposer des contraintes sur la nature des actants dans les schémas de régime des lexèmes. Mais leur position n'est pas clairement définie au sein de la *TST*. On ne sait pas, en particulier, comment ils s'articulent par rapport aux réseaux sémantiques associés aux entrées lexicales, dans le *DEC*.

Un autre enrichissement possible consiste en une définition sémantico-conceptuelle des dépendances syntaxiques profondes, à l'image des cas profonds de [Fillmore, 1968] ou des θ-rôles de la théorie du gouvernement et du liage ([Chomsky, 1981]) ou encore des structures argumentales de [Pustejovsky, 1991] et [Pustejovsky, 1995]. Le remplacement des dépendances syntaxiques profondes par des dépendances sémantico-conceptuelles permet, dans certains cas, d'éviter la duplication des règles de paraphrase. Certaines différences apparaîssant au niveau syntaxique profond dans les structures verbales disparaîssent en effet dans le cadre d'une caractérisation sémantique des actants du verbe. Dans le cas d'un même verbe à l'actif et au passif, l'agent de l'action serait toujours relié au prédicat par une même dépendance. Le remplacement du verbe par un verbe converse serait alors effectué par une même règle, que le verbe soit à l'actif ou au passif.

Mais le remplacement des dépendances syntaxiques profondes par des dépendances sémantico-conceptuelles est difficile à effectuer sans perturber l'équilibre des niveaux de représentation de la *TST* et révèle le statut ambigu des dépendances syntaxiques profondes. Ces dernières sont en effet définies dans la *TST* par rapport aux dépendances syntaxiques de surface : elles sont présentées comme une abstraction de ces dernières. A une dépendance profonde correspond une famille de dépendances de surface. A la dépendance profonde I, par exemple, correspondent les dépendances de surface *sujet grammatical*, telles que *Marie* $\longleftarrow$ *chante* et ses transformées, telles que la dépendance *chant* $\longrightarrow$ *de Marie*. Les critères permettant de regrouper différentes dépendances de surface sous une dépendance profonde unique, et par conséquent le statut des dépendances profondes, ne sont pas clairement identifiés. D'une part, [Mel'čuk, 1988a] (p.63) indique que contrairement aux étiquettes des dépendances sémantiques, qui sont purement distinctives (voir chapitre 2), les dépendances syntaxiques profondes possèdent une sémantique : une dépendance syntaxique profonde regroupe les différentes dépendances syntaxiques de surface réalisant le même argument sémantique d'un prédicat, ce qui leur donne par conséquent une connotation sémantique[6]. D'autre part, la suite du même texte (p.65) met en garde contre l'amalgame que l'on serait tenté de faire entre rôle sémantique et dépendances syntaxiques profondes. On a en effet remarqué à l'occasion de l'application des règles de paraphrase qu'il ne s'agissait pas de dépendances sémantiques, la dépendance numéro I, par exemple, pouvant correspondre aussi bien à un agent qu'à un patient. Le statut des dépendances syntaxique profondes n'est donc, pour le moins, pas clair.

L'introduction de dépendances sémantico-conceptuelles au niveau syntaxique profond suppose donc de modifier la nature de la relation qu'entretient une dépendance syntaxique profonde et les dépendances de surface lui correspondant. La dépendance de surface *sujet*, par exemple, correspondra à la dépendance profonde *agent* dans les cas de constructions actives mais pas dans les cas de constructions passives. L'introduction de dépendances sémantico-conceptuelles nécessite aussi une révision des relations entre les dépendances synaxiques profondes et des dépendances sémantiques (au sens de I. Mel'čuk). De telles modifications dépassent le cadre de cette thèse.

Faisant le constat que les règles de parphrase ne peuvent être effectuées sans faire référence à la sémantique, [Béringer, 1988] propose de les appliquer à un niveau appelé présémantique qui tient du niveau sémantique pour la nature des dépendances et du niveau syntaxique profond pour l'organisation lexicale, ce qui revient a réétiqueter les dépendances du niveau syntaxique profond par des étiquettes sémantiques (au sens de I. Mel'čuk). Cette solution ne résoud néanmoins pas les problèmes de paraphrase illustrés ci-dessus dans la mesure où les dépendances du niveau sémantique sont, comme nous l'avons souligné, dépourvues de signification. Les problèmes que nous avons mis à jour à l'occasion de l'application de règles de paraphrase à des verbes au passif ou à des verbes admettant plusieurs schémas de régime subsistent donc.

Les problèmes soulevés par l'application des règles de paraphrase dans la *TST* illustrent la difficulté d'exploiter les connaissances sémantiques de cette dernière. L'absence de notion de rôle thématique et l'aspect très« *dépouillé*»([Polguère, 1990]) des représentations sémantiques dans le cadre de la *TST* constituent, comme nous venons de le voir, un obstacle au bon fonctionnement du système de paraphrase.

[6]On pourra remarquer que du fait qu'une dépendance de surface ne réalise pas toujours un même actant sémantique, il n'y a pas de raisons pour que le regroupement de plusieurs dépendances de surface réalise un actant sémantique unique.

6.3 Regénération

Nous avons regroupé dans cette section les deux étapes de la regénération distinguées au chapitre 3 : la phase de construction d'une *SSyntS* à partir d'une *SSyntP* transformée et la phase de linéarisation de la nouvelle *SSyntS*. A ce niveau du processus, la majeure partie des choix de reformulation ont déjà été effectués, lors de la phase de transformation de la *SSyntP*. Trois choix paraphrastiques subsistent cependant : la réalisation des fonctions lexicales figurant dans la *SSyntP* transformée, le choix des mots outils, tels que certaines prépositions et les choix de linéarisation de la *SSyntS*. Le premier est effectué lors de la décomposition d'*AEP+* en *AEP*, le second lors du remplacement des *AEP* par des *AES+* ou *AES* et le troisième lors de la linéarisation de la *SSyntS*. Ces choix doivent être faits en fonction des contraintes de la langue contrôlée.

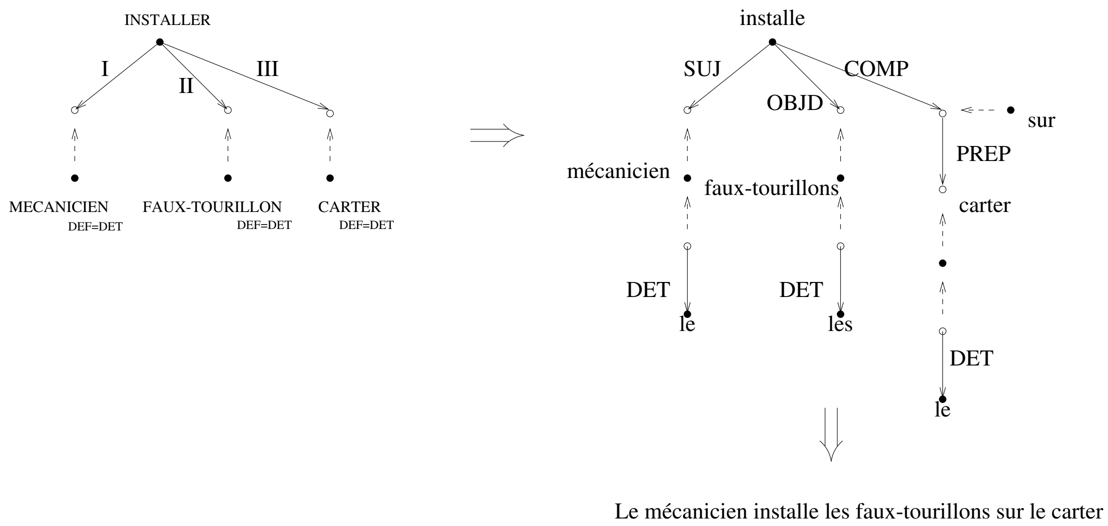


FIG. 6.12: Les deux étapes de la regénération d'une phrase

La problématique de la regénération peut être rapprochée de la génération automatique de texte. Elle s'en distingue néanmoins par deux aspects :

- D'une part, la regénération ne concerne que la partie la plus surfacique de la génération automatique. Le problème de la détermination du contenu informatif de la phrase à générer ne se pose pas : il est identique à celui de la phrase de départ. Notre problématique s'apparente donc plus à la phase de génération dans le cadre de la traduction automatique que [Danlos, 1985] oppose à la discipline de génération automatique de texte, où choix du contenu (quoi dire ?) et production d'un texte (comment le dire ?) doivent être traités. Dans notre cas, le contenu informatif est déjà sélectionné. De plus, les choix importants de lexicalisation ont déjà été effectués, seuls des aspects surfaciques (ordre linéaire, choix de mots outils et valeurs de fonctions lexicales) doivent encore être traités.

- D'autre part, contrairement à la génération automatique, la reformulation procède d'une phrase : la phrase à reformuler. Cette phrase n'est, en général, que partiellement transformée et, à l'issue du processus, certaines parties restent inchangées. Nous avons choisi de ne pas regénérer les parties de la phrase épargnées par la reformulation mais de conserver celles de la phrase d'origine. Les raisons de ce choix sont au nombre de trois.

1. La première a déjà été exposée en 2.2.3 : elle concerne le risque de glissements sémantiques pouvant intervenir à différents moments de la génération de paraphrase. Le problème se pose en particulier pour la valeur des fonctions lexicales : dans certains cas, une fonction lexicale peut prendre pour valeur différents lexèmes qui ne sont pas strictement identiques dans différents contextes. Le choix d'une valeur plutôt que de l'autre peut provoquer des glissements de sens, voire des aberrations. Il est donc plus sûr dans ce cas de choisir la valeur de la phrase initiale.
2. La seconde raison concerne l'incomplétude (voir 2.2.2) des niveaux de représentation mis en jeu dans notre processus de reformulation. En effet, dans le cadre de la *TST*, la tâche de génération à partir du niveau syntaxique profond procède d'une *représentation* syntaxique profonde de la phrase. A ce niveau de représentation certains aspects de la phrase, tels que l'ordre linéaire, ne sont pas représentés explicitement. Ils sont cependant en partie contraints par les informations représentées dans les différentes structures de la représentation syntaxique profonde que sont la *SSyntP*, la structure communicative, la structure anaphorique et la structure prosodique (voir chapitre 2). Dans notre cas, la représentation syntaxique profonde de la phrase que nous manipulons, construite lors de l'analyse, n'est pas complète : elle est réduite à la seule structure syntaxique profonde. Les autres composantes de ce niveau de représentation ne sont pas représentées. Nous ne sommes donc pas en mesure, lors de la génération, d'effectuer ces choix de façon à satisfaire les contraintes communicatives, anaphoriques et prosodiques qui bien qu'existant dans la phrase d'origine n'ont pas été explicitement représentées. Certains glissements indésirables par rapport à la phrase de départ peuvent par conséquent se produire. Là aussi, le risque de glissements peut être limité en préservant certaines informations, telles que l'ordre linéaire de la phrase initiale.
3. Finalement, il ne nous semble pas souhaitable, par respect pour le rédacteur, de transformer des aspects de la phrase initiale qui ne sont pas interdits par la langue contrôlée. Il n'y a en effet pas de raison lorsque le rédacteur a opté pour une formulation particulière autorisée par la langue contrôlée, de la remplacer par une autre.

C'est pour ces trois raisons que nous essayerons autant que possible de ne pas générer, à partir de la *SSyntS* les parties de la phrase qui ne sont pas interdites par la langue contrôlée. Lorsque les choix effectués par le rédacteur dans la phrase de départ ne sont pas interdits par la langue contrôlée, ils seront maintenus. Cette conservation des caractéristiques de la phrase initiale est effectuée par mémorisation des *AES* et *AES+* entrant dans sa composition. Lorsqu'ils sont conformes à la langue contrôlée, ils seront réutilisés dans la phase de regénération.

Pour les trois choix à effectuer que nous avons mentionnés ci-dessus, deux cas sont à distinguer, selon que l'on est en présence d'une situation nouvelle, issue d'une transformation de la *SSyntP* ou d'une situation ancienne, qui n'a pas été perturbée par la transformation de la *SSyntP*. Dans le premier cas, les choix sont effectués uniquement en fonction des contraintes propres à la langue contrôlée. Dans le second, les choix sont effectués en prenant en compte les contraintes de la langue contrôlée et les choix initiaux effectués par le rédacteur.

Le processus de regénération se décompose en deux étapes : une étape de reconstruction de la *SSyntS* à partir de la *SSyntP* transformée et une étape de linéarisation de la *SSyntS* pour donner naissance à une *RMorphP*. Ces deux étapes sont décrites dans les deux sections suivantes.

6.3.1 Reconstruction de la *SSyntS*

La reconstruction de la *SSyntS* se décompose elle-même en deux étapes : une étape de décomposition des *AEP+* entrant dans la constitution de la *SSyntP Analytique*, suivie d'une étape de remplacement des *AEP* par des *AES* ou *AES+*. Lors de la première étape, la valeur des fonctions lexicales syntagmatique est choisie et lors de la seconde, les mots outils sont sélectionnés, d'après les schémas de régime des lexèmes.

Décomposition des *AEP+* en *AEP*

Cette phase est effectuée par application inverse des règles de composition profondes. Ces dernières sont appliquées aux *AEP+* entrant dans la compostion de la *SSyntP Analytique*, les décomposant en *AEP* comme dans l'exemple de la figure 6.13, correspondant à la phrase *Le mécanicien effectue une mesure de tension.*

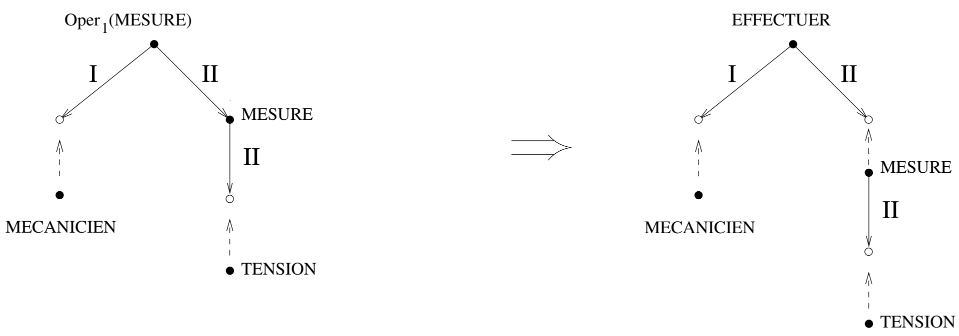


FIG. 6.13: Décomposition des *AEP+*

Les *AEP+* comportent généralement des fonctions lexicales syntagmatiques et, lors de leur décomposition, la valeur des fonctions lexicales doit être choisie. Dans le cas de la figure 6.13, le verbe EFFECTUER a été choisi comme verbe support de MESURE. D'autres valeurs auraient été possibles, telles que PRENDRE. La politique de choix est conforme aux principes évoqués ci-dessus. Si l'on a affaire à un « nouvel » *AEP+*, issu de l'application d'une règle de paraphrase, le choix du verbe support doit se faire en fonction des contraintes de la langue contrôlée : il doit s'agir d'un lexème autorisé. Si en revanche il ne s'agit pas d'un nouvel *AEP+*, le verbe support de la phrase initiale est utilisé.

L'application inverse des règles de composition profonde suppose que ces dernières soient inversibles. Rappelons que l'inversion des règles de composition profondes est toujours possible, du fait de la bijectivité des fonctions de correspondances. L'inversion d'une règle se fait par permutation de sa partie droite et sa partie gauche et inversion de ses fonctions de correspondance. L'inversion du schéma de règle de composition profonde suivant :

$$\begin{gathered} Unif(\mathcal{AEP}_1@\delta\ ,\ \mathcal{AEP}_2@\alpha) \Rightarrow \mathcal{AEP}{+}\ \mid\ f_1\ ,\ f_2 \\ \mathcal{AEP}_1@\beta = FL_{syn}(\mathcal{AEP}_2@\gamma) \end{gathered}$$

est représentée ci-dessous :

$$\begin{gathered} \mathcal{AEP}{+} \Rightarrow Unif(\mathcal{AEP}_1@\delta\ ,\ \mathcal{AEP}_2@\alpha)\ \mid\ f_1^{-1}\ ,\ f_2^{-1} \\ \mathcal{AEP}_1@\beta = FL_{syn}(\mathcal{AEP}_2@\gamma) \end{gathered}$$

Remplacement des *AEP* par des *AES* ou *AES+*

Cette phase est effectuée par application inverse des règles d'abstraction. Comme dans l'étape précédente, deux cas sont à distinguer : le cas où l'on a affaire à un « nouvel » *AEP* et le cas où il s'agit d'un *AEP* laissé intact par la phase de transformation de la *SSyntP* initiale.

Dans le premier cas, les règles d'abstraction inversées sont appliquées, donnant naissance à un ou plusieurs *AES* ou *AES+*. Dans le cas où plusieurs arbres élémentaires sont créés, seuls ceux qui sont autorisés par la langue contrôlée sont sélectionnés. Dans le second cas, l'*AES* ou *AES+* initial est utilisé. Nous avons représenté dans la figure 6.14 un exemple de remplacement des *AEP* d'une *SSyntP Analytique* par des *AES* ou *AES+*. On pourra remarquer que l'*AEP* du verbe EFFECTUER au passé composé a été remplacé par un *AES+* correspondant à la construction auxiliaire, participe passé et que le trait DET = DEFINI associé au substantif MESURE a donné lieu à un article défini.

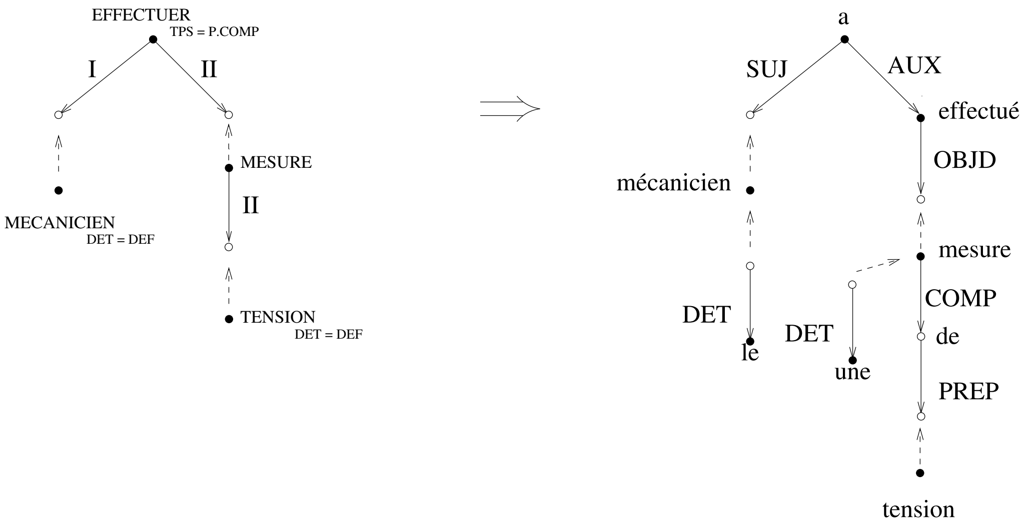


FIG. 6.14: Construction d'une *SSyntS Analytique* à partir d'une *SSyntP Analytique*

6.3.2 Linéarisation de la *SSyntS*

A l'issue de la phase précédente, les *AEP* de la *SSyntP Analytique* ont été remplacés par des *AES*+ ou *AES* autorisés par la langue contrôlée pour donner naissance à une *SSyntS Analytique*. Cette dernière va permettre, par réalisation des opérations d'unification, de composer une *SSyntS* (voir figure 6.15). C'est cette dernière structure qui va être linéarisée à l'étape suivante pour aboutir à une structure morphologique profonde (*RMorphP*) de la phrase.

La linéarisation de la *SSyntS* transformée correspond à la dernière étape de la reformulation. Elle consiste à imposer un ordre total aux nœuds de la *SSyntS* en prenant en compte d'une part l'ordre de la phrase initiale et d'autre part, les contraintes d'ordre propres à la langue contrôlée.

La linéarisation s'effectue en deux temps : dans un premier temps, une structure projective de la *SSyntS* est construite en rattachant les nœuds de la *SSyntS* à leurs gouverneurs linéaires, comme nous l'avons décrit en 4.1.5 ; puis la structure projective est linéarisée. Nous avons représenté dans la figure 6.16 la linéarisation de la *SSyntS* construite à l'étape précédente. Dans cet exemple, la structure à linéariser est projective, la première phase de la linéarisation est donc inutile.

L'algorithme de linéarisation est assez simple de conception : il consiste à parcourir la *SSyntS* à partir de la racine. Pour chaque nœud visité X, la position de ce dernier et de ses fils est déterminée en fonction des contraintes positionnelles associées aux nœuds. Rappelons que les contraintes d'ordre définissent un ordre partiel. Lorsque plusieurs ordres totaux vérifient les contraintes d'ordre, un d'entre eux est choisi au hasard. Le processus est ensuite appliqué récursivement aux fils de X. La simplicité de l'algorithme de linéarisation tient dans le fait que les décisions peuvent être prises de façon locale : l'ordonnancement des nœuds de chaque sous-arbre, à tous les niveaux de l'arborescence, peut être fait indépendamment des sous-arbres frères. Cette propriété

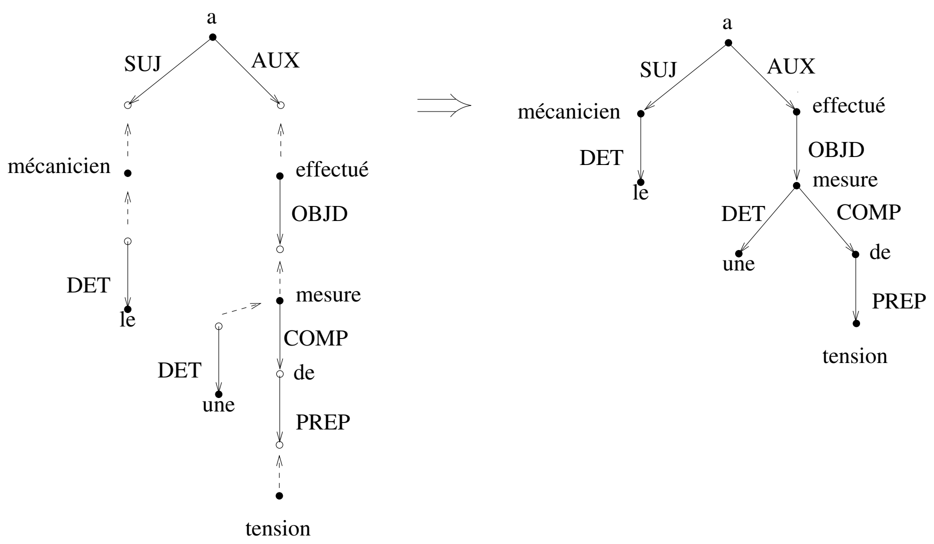


FIG. 6.15: Reconstitution d'une *SSyntS* à partir d'une *SSyntS Analytique*

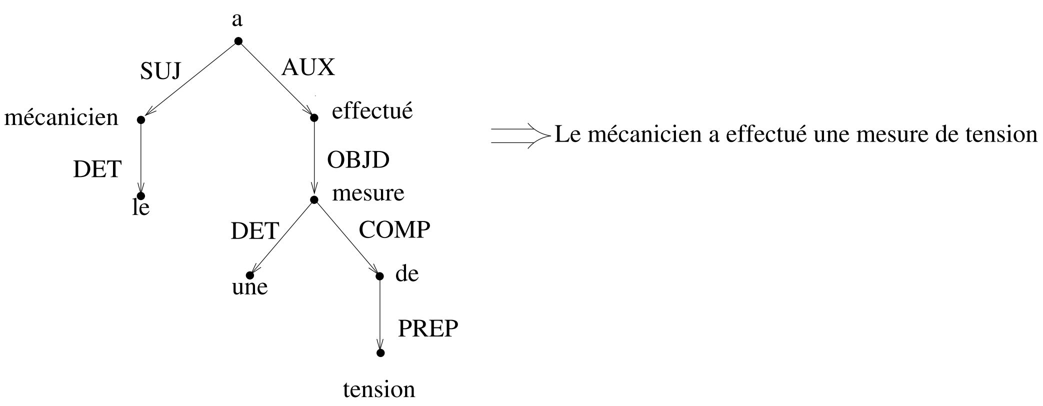


FIG. 6.16: Linéarisation d'une *SSyntS*

est une conséquence de la projectivité de la structure à linéariser. Dans l'exemple de la figure précédente, la racine de l'arbre (le verbe *avoir*) est positionnée par rapport à ses fils (*mécanicien* et *effectué*). Les sous-arbres dominés par ces derniers sont alors linéarisés indépendamment l'un de l'autre.

Pour chaque nœud parcouru, deux cas sont à distinguer : soit il appartient à un *AES* entrant dans la composition de la *SSyntS* initiale, soit il s'agit d'un nouvel *AES*. Dans le premier cas, les informations positionnelles associées au nœud, reflétant l'ordre de la phrase initiale sont verifiées ; si elles sont compatibles avec les règles de la langue contrôlée, alors elles sont utilisées pour le positionnement relatif des deux nœuds, sinon elles sont remplacées par des contraintes conformes. Dans le second cas, lorsque le nœud appartient à un nouvel *AES*, il ne lui correspond pas de contraintes d'ordre provenant de la phrase de départ : le positionnement est effectué en accord avec les contraintes positionnelles autorisées.

6.4 Evolutions par rapport au schéma initial de reformulation

Il devient légitime, après l'ajout des *AES+* effectué au chapitre 5 et celui des *AEP+* dans ce chapitre, de se demander ce qu'il reste du processus initial de reformulation, tel que nous l'avons introduit au chapitre 3. Ce dernier prévoyait deux types d'arbres élémentaires (les *AES* et les *AEP*) et un processus de reformulation basé sur deux types de règles entre arbres élémentaires (les règles d'abstraction et les règles de paraphrase). Dans un premier temps, les *AES* associés aux mots de la phrase étaient combinés entre eux pour former une *SSyntS*. Cette dernière était alors transformée en *SSyntP* lors du remplacement des *AES* par les *AEP* leur correspondant par relation d'abstraction. Certains *AEP* étaient ensuite remplacés par d'autres *AEP* leur correspondant par relation de paraphrase. A ce stade du processus, une nouvelle *SSyntP* était construite, qui allait donner naissance à une nouvelle *SSyntS* puis à une nouvelle phrase, paraphrase de celle de départ. L'ajout de deux nouveaux types d'arbres élémentaires a compliqué le processus qui compte maintenant trois étapes supplémentaires : une étape de regroupement d'*AES* au sein d'*AES+* à l'aide de règles de composition de surface, une étape de regroupement d'*AEP* au sein d'*AEP+*, grâce aux règles de composition profondes et une étape de décomposition d'*AEP+* en *AEP* lors de la phase de regénération.

On peut se demander, au vu de cette inflation, si un tel acroissement de la complexité du processus n'aurait pu être évité et si un modèle de reformulation plus simple et plus conforme à la *TST* n'aurait pu être envisagé[7]. Nous allons tenter de répondre à cette question en rappelant, dans un premier temps, les raisons qui nous ont mené à cette conception du processus de reformulation

[7]Nous ne parlerons pas ici des règles de déplacement introduites au chapitre 5 et des règles d'ajustement introduites dans ce chapitre, car elles n'ont pas de correspondant dans la *TST*. Leur prise en compte dans l'évaluation du modèle et dans sa comparaison avec une approche plus conforme à la *TST* risque par conséquent de fausser les termes de la comparaison.

par manipulation d'arbres élémentaires. Nous reprendrons ensuite les avantages que présente cette approche et les difficultés auxquelles elle s'est heurtée, lors de la mise en correspondance d'*AES* et d'*AEP*, et qui ont mené à la complexité actuelle. Nous nous demanderons alors quelles sont les alternatives possibles à la solution que nous avons choisie.

La notion d'arbre élémentaire de surface avait été motivée par des considérations propres à la tâche d'analyse. Il s'agissait de représenter des contraintes hétérogènes (morphologiques, positionnelles, syntaxiques, lexicales et sémantiques) au sein d'une entité unique en vue d'un traitement automatique plus efficace. Parallèlement, nous avions représenté les règles de paraphrase de la *TST*, à l'image des règles de [Gladkij & Mel'čuk, 1975], sous la forme de correspondances entre arbres élémentaires d'un autre type : les *AEP*. Le lien entre *AES* et *AEP* a ensuite été établi à l'aide de règles d'abstraction, faisant l'hypothèse d'une correspondance directe entre *AES* et *AEP*. Cette correspondance a été une première fois remise en cause au chapitre 5, où la difficulté de mettre en relation *AEP* et *AES*, notamment pour les arbres élémentaires associés à des verbes à temps complexe, a abouti à l'introduction d'un nouveau type d'arbres élémentaires (les *AES+*) et d'un nouveau type de règles (les règles de composition de surface). Nous avons ensuite observé, dans ce chapitre, que les *AEP* obtenus par abstraction des *AES+* ne constituaient toujours pas le niveau de décomposition nécessaire pour l'application des règles de paraphrase, d'où l'introduction d'une nouvelle entité plus étendue (les *AEP+*), et de nouvelles règles (les règles de composition profonde).

Le schéma initial, décrit au chapire 3, constitué d'*AES*, d'*AEP*, de règles d'abstraction et règles de paraphrase présente à nos yeux trois avantages :

- L'existence d'entités complexes, représentant un certain contexte d'occurence d'un mot, que sont les *AES* et *AEP* permet un marquage simple et précis de constructions interdites. Nous avons en effet montré en 3.3 comment, grâce aux *AES* et *AEP*, il est possible d'interdire ou d'autoriser un mot dans un contexte particulier et comment grâce aux notions d'$\mathcal{AES}$ et d'$\mathcal{AEP}$, certaines configurations syntaxiques peuvent être interdites, indépendamment de considérations lexicales. Sans les notions d'*AES*, d'*AEP*, d'$\mathcal{AEP}$ et d'$\mathcal{AEP}$, l'interdiction de certaines configurations syntaxiques et lexico-syntaxiques est plus difficile à énoncer. Elle ne peut plus se faire par marquage direct de certaines entités existantes et nécessite un autre mécanisme de description de phénomènes lexico-syntaxiques pouvant être assez complexes.

- La décomposition de la *SSyntP* en *AEP* (issue du découpage de la *SSyntS* en *AES*), alliée au marquage de certains *AEP* comme interdits permet de se passer de la phase de détection de structures interdites. Lors de la construction de la *SSyntS* et de la *SSyntP*, on sait en effet quels arbres élémentaires interdits ont été utilisés et doivent être remplacés. Sans découpage de la *SSyntP* allié au marquage des *AEP* et *AEP+* interdits, une phase de détection de structures interdites est nécessaire alors que dans notre cas elle est effectuée durant la construction des *SSyntS* et *SSyntP*.

- Le lien entre *AES* et *AEP* permet, lors de la génération d'une nouvelle phrase, de réutiliser des *AES+* correspondant à des *AEP* ou *AEP+* qui n'ont pas été modifiés, comme nous l'avons vu dans la section 6.3.

La première propriété découle de l'existence des *AES* et *AEP*. Les deux secondes sont le résultat de l'existence des arbres élémentaires et des liens d'abstraction et de paraphrase définis entre eux, qui permettent de garder des liens entre les structures manipulées lors du processus de reformulation. C'est pour préserver les liens entre *AES* et *AEP* que nous en somme arrivés à proposer les notions d'*AES+* et d'*AEP+*, en remplaçant l'hypothèse d'un lien direct entre *AES* et *AEP* par un lien indirect, établi par l'intermédiaire des *AES+* et des *AEP+*. L'hypothèse de la correspondance directe entre *AES* et *AEP* a été remplacée par une *hypothèse de composition*, qui considère que les structures nécessaires à un niveau de traitement peuvent être obtenues par composition d'entités du niveau précédent : un *AEP+* peut se décomposer en *AEP* et un *AES+* en *AES*.

L'ajout de trois nouvelles phases dans le processus de reformulation a compliqué le processus, altérant aussi bien ses performances que son économie interne. C'est sur ce dernier point que les conséquences sont le plus importantes : le système s'est « enrichi » de deux nouveaux types de règles (les règles de composition de surface et les règles de composition profonde), ainsi que de deux nouveaux types de structures élémentaires (les *AES+* et les *AEP+*). Le système comporte maintenant plus d'objets élémentaires (nouveaux arbres élémentaires et nouvelles règles), son développement et sa maintenance s'en trouvent alourdis. En particulier, le marquage des structures interdites, qui se limitait à marquer certains lexèmes, *AES*, $\mathcal{AES}$ ou $\mathcal{AEP}$ interdits, doit être étendu au marquage des $\mathcal{AES}$+ et $\mathcal{AEP}$+ interdits. L'influence des phases supplémentaires sur les traitements n'est par contre pas trop pénalisante dans la mesure où le nombre de règles de composition de surface n'est pas très élevé et leur application n'est pas coûteuse. L'analyse du cheminement qui nous a amené à définir les *AES+*, les *AEP+* et les règles de composition a révélé que l'origine de cette inflation résidait dans notre volonté de maintenir un lien entre *AES* et *AEP*. L'alternative la plus naturelle consiste alors à abandonner ce lien. On pourrait maintenir l'existence des *AES* et des *AEP* pour les avantages qu'ils présentent tant au niveau des traitements que comme moyen de marquer des structures interdites dans le lexique et la grammaire, mais par contre abandonner la relation d'abstraction entre *AES* et *AEP*[8]. Le processus serait alors modifié : à l'issue de la construction de la *SSyntS*, la transition vers le niveau syntaxique profond se ferait à l'aide des règles syntaxiques profondes de la *TST* et non plus par substitution d'arbres élémentaires[9]. La transformation de la *SSyntP* se ferait, elle, toujours par substitution d'*AEP*.

[8] La distinction entre *AEP* et *AEP+* ne serait plus pertinente du fait que les *AEP* ne seraient plus définis par abstraction

[9] Il est important de remarquer que les règles de la *TST* permettant d'effectuer la transition *SSyntS* $\longrightarrow$ *SSyntP* sont plus locales que les règles d'abstraction, elles établissent une correspondance entre des éléments structurels moins importants que les arbres élémentaires. Les correspondances sont en effet généralement établies entre une dépendance de surface et une dépendance profonde ou entre deux dépendances de surface et une dépendance profonde, alors que nos règles d'abstraction permettent de remplacer un *AES* entier, pouvant être composé d'un grand nombre de dépendances de surface, par un *AEP*. Le niveau de localité des règles proposé dans la *TST* semble suffisant pour assurer la transition. Nos règles d'abstraction apparaîssent dès lors trop étendues, et masquent un niveau de factorisation qui se contente par conséquent un ensemble plus restreint de règles. Il y a donc plus de règles

Il devient par contre nécessaire, à l'issue de la construction de la *SSyntP*, de partitionner cette dernière en *AEP* et de détecter les *AEP* interdits, afin de les remplacer par des *AEP* autorisés leur correspondant par règle de paraphrase. Ce découpage n'est plus calculé à partir du découpage de la *SSyntS Analytique* en *AES* du fait que la relation entre *AES* et *AEP* n'existe plus. La phase de détection, dont nous nous étions affranchis dans l'approche par arbres élémentaires liés entre eux par relation d'abstraction et de paraphrase, devient nécessaire. Finalement, le dernier avantage que nous avions mis en évidence ci-dessus (la possibilité de réutiliser des *AES* correspondant à des *AEP* non modifiés lors de la phase de transformation de la *SSyntP*) est lui aussi mis à mal par la suppression des règles et des relations d'abstraction. A l'issue de la transformation de la *SSyntP*, les *AEP* qui n'ont pas été modifiés ne sont plus en relation avec les *AES* ayant permi de construire la *SSyntS* initiale lors de l'analyse.

Ce scénario permet donc de s'affranchir des *AES+*, *AEP+* et règles de composition au prix d'une remise en cause des deux derniers avantages exposés ci-dessus. On peut envisager différents critères pour comparer ces deux solutions : l'efficacité, la maintenabilité ou encore la cohérence globale du système. C'est cette dernière que nous avons choisi de privilégier ici. La vision du processus de reformulation à laquelle nous avons abouti nous semble plus satisfaisante du point de vue du modèle. Dans ce dernier, les arbres élémentaires jouent aussi bien un rôle descriptif, en permettant de marquer les structures autorisées et les structures interdites, qu'un rôle opérationnel dans la mise en œuvre du processus.

d'abstraction dans notre approche que de règles de transition de la syntaxe de surface vers la syntaxe profonde dans la *TST*.

Conclusion

Nous avons proposé dans cette thèse un modèle de reformulation fondé sur la Théorie Sens-Texte. Le système a été conçu dans le but de déceler, dans les lignes des manuels d'entretien et de fonctionnement de systèmes aéronautiques, des formulations proscrites par une langue contrôlée, pour en proposer d'autres conformes à la norme. Notre travail a consisté à élaborer un formalisme de représentation des connaissances linguistiques nécessaires à la tâche de reformulation ainsi que les différents modules de réalisation des étapes de la reformulation.

Le formalisme a été élaboré en prenant en compte deux contraintes majeures. La première concerne l'intégration, dans un même formalisme, de connaissances linguistiques générales et de règles d'écriture d'une langue contrôlée. L'idée consiste à représenter, au sein d'un même langage formel, les connaissances linguistiques de la *TST* permettant de générer plusieurs paraphrases d'une phrase donnée et de certaines règles d'écriture d'une langue contrôlée qui peuvent être vues comme des règles de préférence stytlistiques, guidant la génération de paraphrases. D'autre part, et c'est la seconde contrainte, le formalisme proposé doit permettre une mise en œuvre automatique efficace du processus de reformulation. Ces deux contraintes nous ont amené à proposer un formalisme assez différent des règles de la *TST*. Celui-ci est en effet basé principalement sur la notion d'arbres élémentaires, se rapprochant ainsi des grammaires d'arbres adjoints lexicalisées (LTAG) et des dendrogrammaires de A. Gladkij et I. Mel'čuk. L'existence d'arbres élémentaires qui représentent, sous la forme d'un sous-arbre lexico-syntaxique, un contexte d'occurrence d'un mot donné, permet de prendre en compte les règles d'écriture d'une langue contrôlée. Il suffit en effet d'identifier, parmi les arbres élémentaires constituant la grammaire du système, les arbres décrivant une construction lexico-syntaxique interdite par la langue contrôlée. Le but du processus de reformulation est alors d'éliminer, dans la représentation syntaxique d'une phrase, les arbres élémentaires interdits, en les remplaçant par des arbres élémentaires autorisés, grâce aux capacités paraphrastiques de la *TST*. Le formalisme initial a été présenté dans le chapitre 3, il comporte deux types d'arbres élémentaires : les arbres élémentaires de surface et les arbres élémentaires profonds qui se distinguent par leur niveau d'abstraction, les arbres profonds étant plus abstraits que les arbres de surface. Deux types de relations peuvent lier les arbres élémentaires entre eux : la relation d'abstraction, qui permet de lier un arbre élémentaire profond et un arbre élémentaire de surface et la relation de paraphrase, permettant de lier deux arbres élémentaires profonds. Les relations d'abstraction et de paraphrase sont établies par des règles de même nom. Des contraintes de mise en œuvre du processus de reformulation nous ont amené à enrichir le formalisme en introduisant, dans les chapitres 5 et 6, de nouveaux arbres élémentaires, les arbres élémentaires étendus et de nouveaux types de règles, les règles de composition, de déplacement et d'ajustement. Le formalisme s'est donc considérablement enrichi. Il reste néanmoins constitué des mêmes éléments de base : des arbres élémentaires et des règles permettant d'établir des relations entre arbres élémentaires.

Le processus de reformulation peut être rapproché, dans sa conception, d'une architecture de traduction automatique basée sur le transfert, il se déroule en trois étapes : une étape d'analyse de la phrase initiale, permettant de construire une structure syntaxique profonde lui correspondant, une étape de transformation de la structure syntaxique profonde et une phase de regénération d'une phrase à partir de la structure syntaxique profonde transformée. Les trois étapes sont réalisées par manipulations d'arbres élémentaires, effectuées grâce à deux opérations : une opération de composition d'arbres élémentaires de surface, appelée opération d'attachement et une opération dite de substitution, permettant le remplacement d'un arbre élémentaire par un autre au sein d'une structure syntaxique plus étendue. La phase d'analyse consiste à composer par attachement les arbres élémentaires de surface correspondant aux mots de la phrase initiale afin de construire la structure syntaxique de surface de cette dernière. Les différents arbres élémentaires de surface entrant dans la composition de la structure syntaxique de surface créée sont alors remplacés par leur correspondant profond grâce à l'opération de substitution, pour constituer la structure syntaxique profonde de la phrase initiale. Les arbres élémentaires de la structure syntaxique profonde marqués comme interdits au regard d'une langue contrôlée sont remplacés lors de la phase de transformation par d'autres arbres élémentaires profonds, autorisés par la langue contrôlée, avec lesquels ils sont liés par relation de paraphrase. A ce stade, une structure syntaxique profonde exempte d'arbres élémentaires interdits a été créée ; elle constitue le point de départ de la phase de regénération, qui permet de créer une nouvelle phrase. La phase de génération se décompose en deux étapes : une étape de construction d'une structure syntaxique de surface par application inverse des règles d'abstraction puis une étape de linéarisation de cette structure. La phase de regénération se fait en tenant compte de deux contraintes : la conformité aux règles de la langue contrôlée et le respect de la phrase initiale. Ainsi, les éléments de la phrase initiale qui ne sont pas interdits par la langue contrôlée seront conservés dans la nouvelle phrase. L'introduction des arbres élémentaires étendus et des nouvelles règles a quelque peu compliqué le processus, sans toutefois en modifier le principe général.

La description rapide du modèle que nous venons d'effectuer suscite deux questions. La première, la plus naturelle, est la suivante : le modèle proposé permet-il de générer des reformulations qui soient à la fois correctes sur le plan grammatical, conformes à la langue contrôlée et paraphrastiques de la phrase initiale ? La seconde question concerne l'adéquation du modèle proposé : définit-il les « bons » éléments ? sa complexité est-elle une conséquence de la complexité du problème traité, ou est-elle le fruit d'une mauvaise adéquation du modèle au problème traité ?

Nous avons tenté de répondre à la seconde question en 6.4, après avoir constaté la nécessité d'introduire de nouveaux éléments dans le formalisme et dans les traitements et après avoir analysé les raisons de cet accroissement de la complexité du modèle. Celle-ci provient principalement de la notion de relation d'abstraction définie entre arbres élémentaires de surface et arbres élémentaires profonds. Nous avons montré que cette relation est un maillon important de la chaîne de reformulation et que sa présence garantit les principaux avantages de notre modèle de reformulation, il ne nous a pas semblé possible de conserver ces avantages en faisant l'économie de la relation d'abstraction. Le niveau de complexité du modèle semble par conséquent inhérent au principe de reformulation que nous avons adopté, qui reste séduisant à nos yeux.

La réponse à la première question est plus délicate car il est difficile d'évaluer un modèle qui est incomplet et le notre l'est de deux points de vue : d'une part, comme nous le mentionnions dans la présentation, notre volonté de proposer un modèle de tout le processus de reformulation nous a obligé à délaisser certains aspects de notre problématique ; d'autre part, l'évaluation du modèle ne peut être menée sans l'implémentation d'une grammaire à large couverture et de nombreuses règles de paraphrase. Nous allons nous pencher sur ces deux aspects dans les paragraphes suivants.

Les différents aspects de la problématique de la reformulation qui n'ont pas été pris en compte dans notre modèle ne seront pas ennumérés ici. L'attention sera portée sur trois d'entre eux qui nous semblent cruciaux :

- Le premier problème réside dans la définition des langues contrôlées. Comme nous l'avons évoqué au chapitre 1, les langues contrôlées dans leurs versions actuelles sont souvent incomplètes et peu précises. Nous avons fait l'hypothèse que les règles d'une langue contrôlée peuvent être« traduites» en termes plus linguistiques pour être introduites dans notre modèle sous la forme d'une distinction entre structures interdites et structures autorisées. L'inadéquation des règles des langues contrôlées, dans leur forme actuelle, pour le traitement automatique est un problème auquel toute tentative d'automatisation du processus de reformulation en langue contrôlée est confronté. Nous pensons avoir proposé, grâce aux arbres élémentaires et au types d'arbres élémentaires, des unités de description assez souples pour représenter des phénomènes lexico-syntaxiques complexes et variés, qui pourront servir de base à une représentation lexico-syntaxique des règles d'une langue contrôlée.

- La *TST*, à laquelle nous avons emprunté un bon nombre de concepts et d'hypothèses, comporte, comme nous l'avons vu en 2.2.3 et en 6.2 un certain nombre d'imperfections qui ne garantissent, en particulier, ni la synonymie entre phrases engendrées par son système de paraphrase ni leur grammaticalité. Il s'agit là de problèmes linguistiques théoriques importants et nous avons vu qu'une partie d'entre eux nécessitait la prise en compte de connaissances sémantiques et conceptuelles qui sont mal décrites voire absentes dans la *TST*. Ces problèmes illustrent la complexité de la tâche de reformulation qui met en jeu des aspects de la linguistique encore mal connus.

- Finalement, le modèle de reformulation que nous avons proposé se restreint, dans les différents niveaux de représentation de la *TST* qu'il met en œuvre, aux structures principales de ces derniers (structure morphologique profonde, structure syntaxique de surface et structure syntaxique profonde). Ils délaissent les structures anaphoriques, prosodiques et communicatives. Or il est clair que ces denières structures sont nécessaires pour garantir une reformulation de qualité. Leur prise en compte n'est cependant pas une chose simple : elle suppose d'une part que ces structures soient mieux décrites dans la *TST* et d'autre part qu'elles soient intégrées dans notre formalisme et prises en compte lors des traitements. Un tel programme, par son ampleur, dépassait les limites de cette thèse.

D'autre part, nous n'avons pas proposé une implémentation à large couverture dans notre formalisme. Il est probable que la prise en compte de phénomènes lexico-synaxiques nouveaux révèle certaines faiblesses du formalisme et nécessite un enrichissement de ce dernier. Néanmoins, le fait de nous être appuyé sur la *TST* et sur des formalismes grammaticaux existant, tels que les grammaires d'arbres adjoints lexicalisées, offrent une certaine garantie quant à la possibilité de prendre en compte un large éventail de phénomènes linguistiques.

ANNEXES

Annexe A

Arbres élémentaires et règles

Différents arbres élémentaires et règles ont étés introduits tout au long de la thèse. Certains ont été définis dès l'introduction du modèle de reformulation, au chapitre 3, il s'agit des éléments fondamentaux du modèle, permettant la mise en œuvre de la reformulation telle que nous l'envisageons. D'autres arbres élémentaires et règles ont ensuite été introduits, dans différents chapitres pour résoudre certains problèmes locaux rencontrés lors de différentes phases de la reformulation.

A.1 Les arbres élémentaires

Les arbres élémentaires constituent l'unique moyen de représentation des structures lexico-syntaxiques nécessaires à la reformulation. A tout arbre élémentaire correspond un type d'arbre élémentaire, qui peut être vue comme la structure syntaxique auquel correspond l'arbre élémentaire. Ce dernier est alors défini par la réunion d'un type d'arbre élémentaire et d'un lexème. Les arbres élémentaires ne sont par conséquent pas représentés dans la grammaire qui est constituée de types d'arbres élémentaires. Les arbres élémentaires sont créés dynamiquement lors du processus de reformulation.

1. Les arbres élémentaires de surface (*AES*)

   Les *AES* ont été introduits au chapitre 3. Ils jouent deux rôles majeurs dans le processus de reformulation, un rôle vis-à-vis de l'analyse en définissant les structures élémentaires du processus de construction de la *SSyntS* (chapitre 5) et un rôle de description de structures lexico-syntaxiques qui sont interdites ou autorisées par une langue contrôlée (chapitre 3).

2. Les arbres élémentaires de surface étendus (*AES+*)

   Les *AEP+* ont été introduits au chapitre 5 comme regroupement de plusieurs *AES*. Ce regroupement, effectué grâce aux règles de composition de surface, permet d'établir un lien indirect entre *AES* et *AEP*. En effet, lorsqu'un *AES* ne peut être mis en correspondance directe avec un *AEP*, il est regroupé avec d'autres *AES* au sein d'un *AES+* et ce dernier est mis en correspondance, grâce à une règle d'abstraction, avec un *AEP*.

3. Les arbres élémentaires profonds (*AEP*)

Les *AEP* ont été introduits au chapitre 3 comme unité structurale sur lesquelles appliquer les règles de paraphrase. Ils permettent aussi d'identifier certaines constructions lexico-syntaxiques interdites au regard d'une langue contrôlée. Les *AEP* sont de plus liés à des *AES* ou à des *AES+* grâce aux règles d'abstraction. Ils sont doivent donc répondre à deux exigences, correspondre aux bonnes unités d'application des règles de paraphrase et correspondre structurellement à des *AES* ou *AES+* pour pouvoir être mis en relation d'abstraction avec ces derniers.

4. Les arbres élémentaires de surface étendus (*AEP+*)

   Dans certains cas, les *AEP* définis par abstraction d'*AES* ou d'*AES+* ne correspondent pas aux *AEP* définis pour les besoins de la paraphrase. Plusieurs *AEP* sont alors regroupés, grâce à des règles de composition profondes au sein d'un *AEP+*, c'est sur ce dernier que pourront être appliquées des règles de paraphrase.

A.2 Les règles

Les règles permettent d'effectuer les diverses manipulations d'arbres nécessaires pour la mise en œuvre du processus de reformulation. Nous avons distingué deux types de règles, les règles élémentaires et les règles complexes, les dernières correspondant à un regroupement de plusieurs règles élémentaires. Les différentes règles sont définies au niveau des types d'arbres élémentaires, pour s'appliquer sur des instances d'arbres élémentaires.

A.2.1 Les règles élémentaires

Les règles élémentaires permettent d'effectuer trois types d'opération sur les arbres élémentaires, l'établissement de relations entre arbres élémentaires, le regroupement d'arbres élémentaires et le déplacement d'un arbre élémentaire. Ces trois opérations sont effectuées grâce aux trois types de règles présentées ci-dessous :

1. Les règles de correspondance

   Les règles de correspondance s'appliquent sur un arbre élémentaire pour en créer un nouveau tout en établissant des liens entre les deux arbres. Les liens sont matérialisés par une fonction dite fonction de correspondance, définie entre les ensembles des adresses de chacun des deux arbres élémentaires. Les règles de correspondances peuvent être enrichies de conditions, représentées sous la règle elle-même. Trois types de règles de correspondance ont été définies :

   (a) Les règles d'abstraction

   $$\mathcal{AES}{+} \Rightarrow \mathcal{AEP} \mid f$$

   Les règles d'abstraction permettent de créer, à partir d'un *AES* ou d'un *AES+*, un *AEP* et d'établir une relation d'abstraction entre les deux. Les règles d'abstraction entre un *AES* et un *AEP* ont été introduites au chapitre 3, elles ont été élargies aux *AES+*, après l'introduction de ces derniers, au chapitre 5.

(b) Les règles de paraphrase

$$\mathcal{AEP}{+}_1 \Rightarrow \mathcal{AEP}{+}_2 \mid f$$

$$\mathcal{AEP}{+}_1@X = FL_{par}(\mathcal{AEP}{+}_2@Y)$$

Les règles de paraphrase correspondent au regroupement des règles lexicales de paraphrase et des règles syntaxiques de paraphrase de la *TST*. Elles permettent de créer, à partir d'un *AEP*, un autre *AEP*, paraphrastique du premier et de relier leurs nœuds grâce à une fonction de correspondance. Les règles de paraphrase mettent en jeu une fonction lexicale paradigmatique liant un nœud du premier arbre à un nœud du second. Lors de l'application d'une règle de paraphrase à un *AEP*, rien ne garantit l'existence, dans le lexique, de la fonction lexicale sur laquelle repose la règle, c'est pourquoi l'existence de la fonction lexicale est ajoutée à la règle en tant que condition. Les règles de paraphrase ont été introduites au chapitre 3 entre $\mathcal{AEP}$. Elles ont été étendues aux $\mathcal{AEP}+$ dans le chapitre 5 pour pouvoir lier deux *AEP* entre eux, deux *AEP+* entre eux ou un *AEP* et un *AEP+*.

(c) Les règles de dérivation

$$\mathcal{AEP}_1 \Rightarrow \mathcal{AEP}_2 \mid f$$
$$\mathcal{AEP}_1@\alpha = FL_{par}(\mathcal{AEP}_2@\beta)$$

Les règles de dérivation ont été introduites au chapitre 6 pour effectuer certaines transformations lexico-syntaxiques telles que la transformation d'un *AEP* adjectival à un *AEP* adverbial. Ces transformations sont nécessaires, lors de l'application de règles de paraphrase pour garantir la bonne forme des modificateurs et permettre le bloquage de certaines paraphrases. Elles mettent en jeu une fonction lexicale paradigmatique entre un lexème de chaque arbre élémentaire, tel que la fonction permettant de relier un nom à un adjectif ou un adverbe dérivés.

2. Les règles de composition

Les règles de composition ont été introduites à la suite de la définition des arbres élémentaires étendus aux chapitres 5 et 6. Elles permettent de créer, à partir de deux arbres élémentaires ou plus, un arbre élémentaire étendu et de lier, grâce à des fonctions de correspondance, les nœuds des arbres élémentaires aux nœuds de l'arbre élémentaire étendu. Les arbres élémentaires à partir desquels est créé l'arbre étendu doivent être liés entre eux au sein d'une structure syntaxique analytique. Nous représenterons le lien entre deux arbres A_1 et A_2 dans une structure analytique par l'expression $Unif(A_1@X, A_2@Y)$ représentant le fait qu'il existe un arc, dans l'arbre syntaxique analytique entre le nœud d'adresse X de A_1 et le nœud d'adresse Y de A_2. Deux types de règles de composition ont été définies :

(a) Les règles de composition de surface

$$Unif(\mathcal{AES}_1@X,\ \mathcal{AES}_2@Y) \Rightarrow \mathcal{AES}+ \ \mid \ f_1,\ f_2$$

Les règles de composition de surface ont été introduites au chapitre 5. Elles permettent de créer à partir de deux *AES* liés entre eux dans un arbre syntaxique de surface analytique, un *AES+* et de lier, par l'intermédiaire de deux fonctions de correspondance, les nœuds des *AES* aux nœuds de l'*AES+*. Les règles de composition profonde peuvent être étendues, si besoin en est, à la composition de plus de deux *AES*.

(b) Les règles de composition profonde

$$Unif(\mathcal{AEP}_1@W, \mathcal{AEP}_2@X) \Rightarrow \mathcal{AEP}+ \ \mid \ f_1,\ f_2$$

$$\mathcal{AEP}_1@Y = FL_{syn}(\mathcal{AEP}_2@Z)$$

Les règles de composition profondes permettent de créer à partir de deux *AEP* liés entre eux dans un arbre syntaxique profond analytique, un *AEP+* et de lier, par l'intermédiaire de deux fonctions de correspondance, les nœuds des *AEP* aux nœuds de l'*AEP+*. Les règles de composition profondes permettent de plus de déceler dans la structure syntaxique les phénomènes de collocation représentées, dans la *TST*, par des fonctions lexicales syntagmatiques. La condition représentée sous la règle impose en effet l'existence, dans le lexique, d'une fonction lexicale syntagmatique reliant le lexème porté par un nœud du premier *AEP* ($\mathcal{AEP}_1@Y$) et le lexème porté par un nœud du second *AEP* ($\mathcal{AEP}_2@Z$).

3. Les règles de déplacement

$$Unif(\mathcal{AEP}_1@X\ ,\ \mathcal{AEP}_2@0) \Rightarrow Unif(\mathcal{AEP}_1@Y\ ,\ \mathcal{AEP}_2@0)$$

Les règles de déplacement sont les seules règles ne provoquant pas la création d'un nouvel arbre élémentaire. Elles se bornent en effet à déplacer, dans une structure syntaxique analytique le site d'attachement d'un arbre élémentaire sur un autre. La règle présentée ci-dessus permet de changer le site d'attachement d'AEP_2 sur AEP_1. AEP_2 qui était attaché au nœud $AEP_1@X$ se trouve attaché, après application de la règle au nœud $AEP_1@Y$. Les règles de déplacement ont été introduites au chapitre 5 pour déplacer certains *AES* ou *AES+* avant l'application des règles d'abstraction et chapitre 6 au sein des règles d'ajustement.

A.2.2 Les règles complexes

Les règles complexes sont composées de plusieurs règles élémentaires. La décomposition des règles complexes en règles élémentaires permet d'utiliser une règle élémentaire dans plusieurs règles complexes différentes. Deux types de règles élémentaires ont été définies :

1. Les règle d'ajustement

$$Unif(\mathcal{AEP}_1@\alpha\ ,\ \mathcal{AEP}_2@0) \Rightarrow Unif(\mathcal{AEP}_1@\beta\ ,\ \mathcal{AEP}_3@0)$$
$$\mathcal{AEP}_3 = \mathcal{R}d\acute{e}riv(\mathcal{AEP}_2)$$

   Les règles d'ajustement sont composées d'une combinaison d'une règle de dérivation et d'une règle de déplacement. Elles ont été introduites au chapitre 6 pour rectifier certaines incohérences apparaissant à la suite de l'application d'une règle de paraphrase. Dans l'exemple présenté ci-dessous, $\mathcal{AEP}_1$ est transformé par une règle de dérivation en $\mathcal{AEP}_3$ et son site d'attachement sur AEP_1, est modifié, passant du nœud d'adresse X au nœud d'adresse Y.

2. Les règles de paraphrase généralisées

   Règle généralisée de paraphrase:

   Règle de paraphrase
   {Règle d'ajustement1}
   {Règle d'ajustement2}
   ...

   Les règles de paraphrase généralisées sont composée d'une règle de paraphrase et d'une ou plusieurs règles d'ajustement. Elles permettent d'effectuer simultanément la paraphrase et les ajustements qu'elle nécessite. Les règles d'ajustement jouent aussi le rôle de condition d'application de la règle de paraphrase, si leur application échoue, suite, par exemple à l'échec d'une fonction lexicale, la règle de paraphrase échoue aussi.

Annexe B

Grammaire d'analyse et règles de paraphrase

Nous allons décrire brièvement dans cette annexe quelques structures grammaticales que nous avons implémentés dans notre formalisme. Nous nous limiterons à la description de quelques *AES* et de quelques règles de parphrase. L'ensemble des *AES* constitue la grammaire d'analyse et les règles de paraphrase déterminent les capacités paraphrastiques du modèle. Les autres éléments du formalisme, *AES+*, *AEP+*, règles de composition, de déplacement et d'ajustement, sont définies en fonction des *AES* et des règles de paraphrase. Il est important de noter que les *AES* et les règles de paraphrase décrits dans cette annexes ne sont là qu'à titre d'illustration, les *AES* proposés ne décrivent que quelques structures lexico-syntaxiques et les règles de paraphrase ne donnent qu'un échantillon de règles possibles.

B.1 Les arbres élémentaires de surface

La grammaire d'analyse consiste en un ensemble d'$\mathcal{AES}$ et de liens établis, dans le dictionnaire lexical entre chaque lexème et les $\mathcal{AES}$ qu'il peut ancrer, pour former des *AES*. La morphologie des $\mathcal{AES}$ dépend en grande partie de la définition des dépendances syntaxiques qui entrent dans leur composition. Des jeux différents de dépendances syntaxiques induiront des $\mathcal{AES}$ différents et donc des grammaires différentes. S'il existe un consensus sur certaines dépendances, telles que les dépendances *sujet* et *objet direct*, d'autres, telles que les dépendances relatives à la coordination sont définies différemment selon les auteurs. L'écriture d'une grammaire doit par conséquent débuter par la définition d'un ensemble de dépendances syntaxiques. Nous ne suivrons pas ici cette voie car la définition d'un jeu de dépendances syntaxiques est un travail important en soi, compliqué par l'inexistence de critères universels de définition des dépendances syntaxiques. Nous nous inspirerons ici les dépendances définies dans [Béringer, 1988], elles mêmes basées sur les dépendances définies dans le cadre d'une grammaire de l'anglais dans [Mel'čuk, 1988b].

Les *AES* décrits ci-dessous vérifient les trois principes de bonne formation des *AES* proposés au chapitre 4. En particulier, chaque *AES* ne possède qu'une ancre lexicale, représentée graphiquement par un nœud noir. Un *AES* peut toutefois posséder plusieurs nœuds lexicaux, mais seul un d'entre eux possède le statut d'ancre lexicale. Les *AES* sont représentés dans notre modèle par des structures de traits complexes. Nous avons préféré les représenter ici, pour des raisons de lisibilité, sous forme graphique. De plus, nous nous intéresserons principalement à l'aspect suructurel des *AES*, délaissant les contraintes morphologiques positionnelles et sémantiques existant au sein des *AES*. Les nœud seront étiquetés par une catégorie syntaxique ou par un lexème lorsqu'il s'agit d'un nœud lexicalement contraint.

La grammaire, telle qu'elle est implémentée actuellement est plate, il n'existe pas de moyens de regrouper des $\mathcal{AES}$ en familles ni de dériver certains $\mathcal{AES}$ à partir d'autres à l'aide de règles lexicales. C'est là un manque et la structuration d'une grammaire devient très vite nécessaire à sa maintenance et à son enrichissement. Nous allons décrire ci-dessous quelques *AES* définis dans la grammaire selon la catégorie de leur ancre lexicale, c'est en effet par l'intérmédiaire de cette dernière que les arbres sont indexés dans le dictionnaire lexical.

B.1.1 Les verbes

Les verbes peuvent ancrer des *AES* différents selon qu'il sont participes passés, participes présents, à l'infinitif ou conjugués à un temps fini. Nous commencerons par décrire les *AES* associés aux verbes à temps simples avant décrire les *AES* associés aux participes passés.

Les verbes à temps simple

Nous distinguerons les *AES* ancrés par des verbes ordinaires de ceux ancrés par des auxiliaires de temps ou des verbes modaux.

- Les verbes ordinaires

  Les verbes ordinaires ancrent des *AES* de racine verbale. Les arguments du verbe, qu'ils soient nominaux, prépositionnels ou phrastiques sont représentés comme dépendants directs ou indirects du verbe. Nous avons représenté dans la figure B.1 les *AES* associés aux verbes *vivre, manger, donner à* et *penser que.*

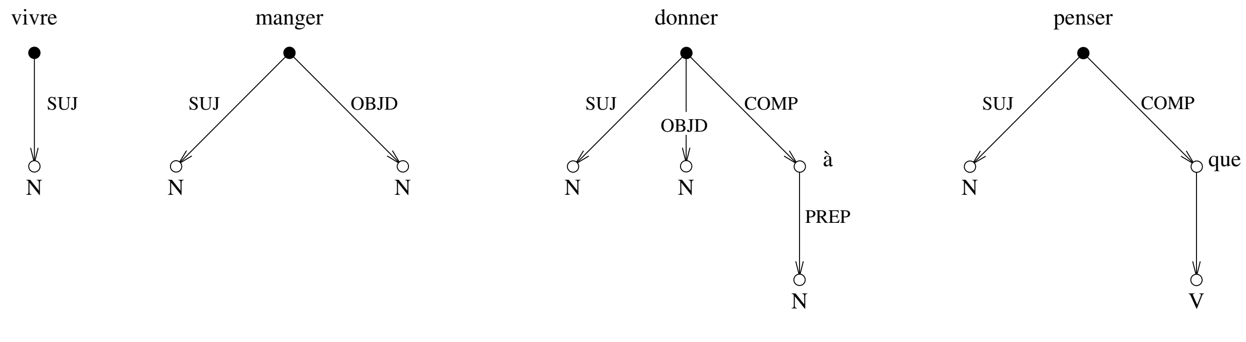


FIG. B.1: Verbes ordinaires

- Les auxiliaires de temps

  Les auxiliaires de temps sont représentés par des *AES* de racine verbale. Cette dernière possède deux dépendants, le participe passé et le sujet.

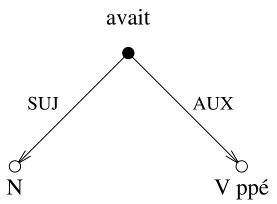


FIG. B.2: Auxiliaires de temps

- Les verbes modaux et aspectuels

  Les *AES* associés aux verbes modaux et aspectuels ressemblent à ceux des auxiliaires de temps, une première dépendance lie le verbe au sujet et une seconde lie le verbe à un infinitif ou à une préposition, elle-même liée à l'infinitif. Nous avons représenté dans la figure B.3 les *AES* des verbes *pouvoir* et *venir de.*

FIG. B.3: Verbes modaux et aspectuels

- Les verbes copules

  Les verbes copules ancrent aussi des *AES* à racine verbale à laquelle sont liés le sujet et l'attribut.

Les participes passés

Les participes passés ancrent des *AES* différents selon qu'ils sont participes passés de verbes à temps composés, de verbes de phrases passives ou en position d'épithètes.

- Les participes passés de verbes à temps composés

  Les participes passés de verbes à temps composés ancrent des *AES* dont la racine est constituée par le nœud correspondant à l'auxiliaire. Les participes passés admettant le verbe *être* et

FIG. B.4: Verbes copules

ceux admettant le verbe *avoir* comme auxilaires n'étant pas soumis aux mêmes contraintes d'accord, ancrent des *AES* différent. Le participe passé pouvant imposer des contraintes sémantiques ou morphologiques au sujet, ce dernier est inclu dans l'arbre élémentaire. Les *AES* ancrés par un participe passé peuvent être vus comme les *AES* des mêmes verbes à temps fini auxquels on aurait dédoublé la racine et redistribué les actants entre l'auxiliaire et le participe passé, en laissant au niveau du premier le sujet et en descendant les autres au niveau du participe passé.

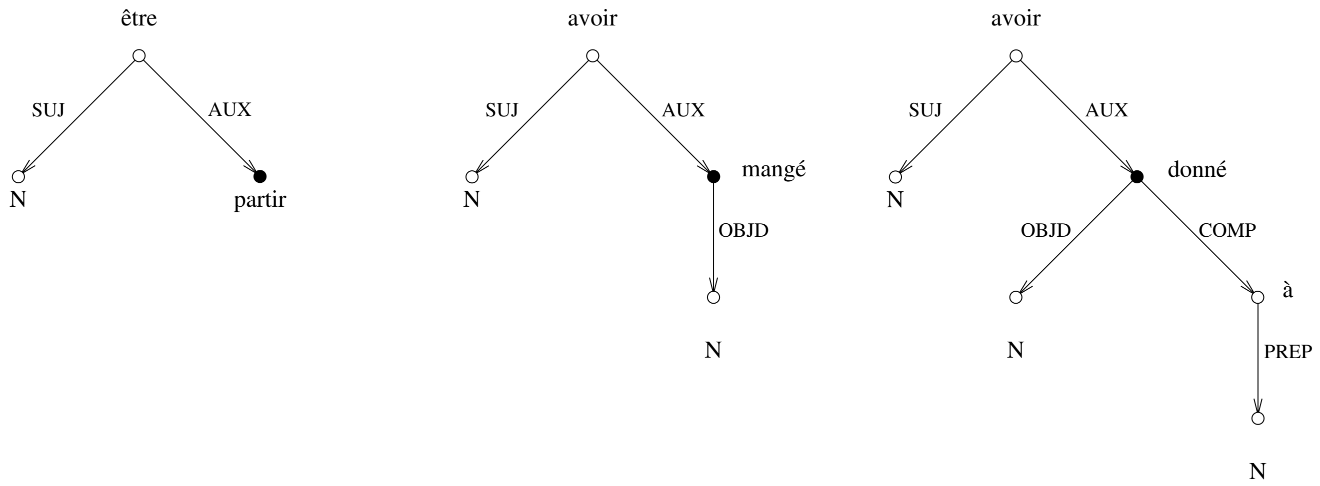


FIG. B.5: Participe passé de verbe à temps composé

- Les participes passés de verbes passifs

  Les participes passés de verbes passifs ancrent des *AES* à la racine desquels est représenté l'auxiliaire. Le complément d'agent est introduit par une dépendance *agent* au niveau du participe passé.

- Les participes passés en fonction épithète

  Les participes passés en fonction épithète ancrent des *AES* semblables à ceux des adjectifs occupant la même fonction. La racine de l'*AES* est constituée d'une description des noms modifiés par l'adjectif. La présence du nom au sein de l'*AES* permet d'imposer des

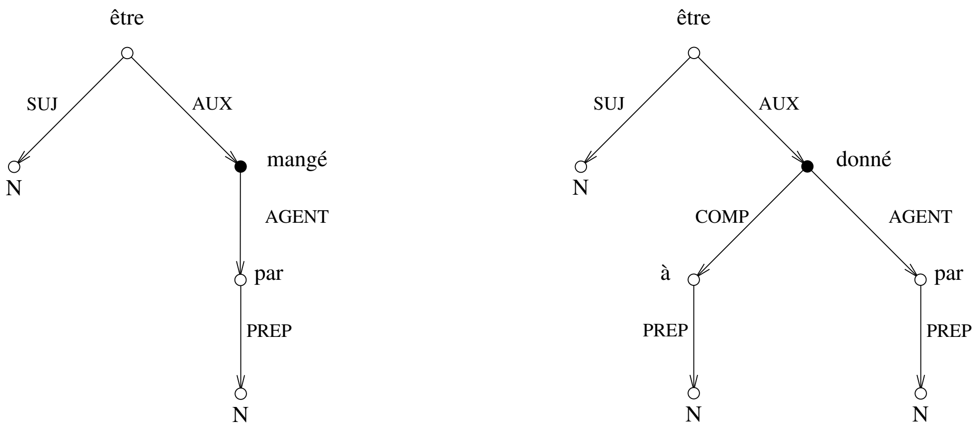


FIG. B.6: Participe passé de verbes au passif

contraintes lexico-sémantique sur ce dernier. Les actants éventuels du participe passé sont représenté comme dépendants de ce dernier.

FIG. B.7: Participe passé en fonction épithète

B.1.2 Les prépositions

Les prépositions peuvent ancrer deux types d'*AES* différents selon qu'elles introduisent un actant ou un circonstant. Dans le premier cas, l'*AES* est réduit au nœud portant la préposition. En effet, dans ces cas la préposition existe déjà dans l'*AES* correspondant à l'unité prédicative dont la préposition introduit un actant. Lorsque la préposition introduit un circonstant ou un modifieur, elle ancre un *AES* plus complexe dont la racine est composée de la description du lexème qu'elle modifie. Aux prépositions pouvant introduire des actants ou des modifieurs seront associés les deux *AES*.

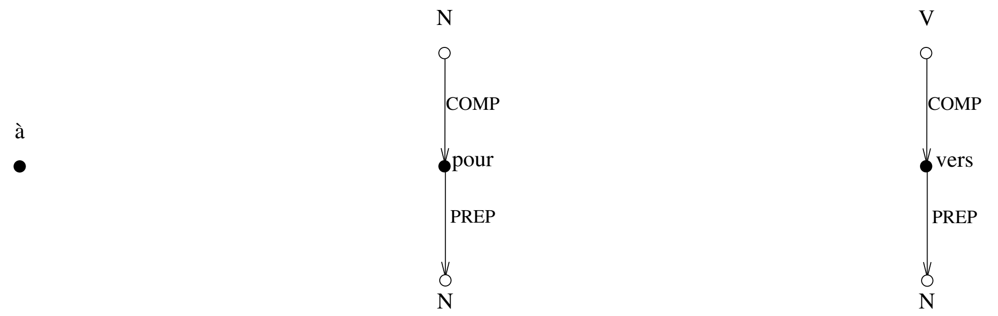


FIG. B.8: Prépositions

B.1.3 Les noms

Les noms sont représentés par des *AES* dont ils constituent la racine. Lorsque le nom possède des actants, ces derniers sont représentés dans l'*AES* comme dépendants du nom.

FIG. B.9: Noms simples et noms prédicatifs

B.1.4 Les articles

Les articles sont représentés comme des ajouts au nom, ils ancrent des *AES* dont la racine est constituée du nom qu'ils déterminent.

FIG. B.10: Articles

B.1.5 Les adjectifs

Les adjectifs ancrent des *AES* différents selon qu'ils sont épithètes ou attributs.

- Les étpithètes

  Les épithètes sont représentés par des *AES* ayant pour racine un nœud représentant le nom modifié. Les éventuels actants des adjectifs sont représentés dans l'*AES*.

FIG. B.11: Adjectifs en position d'épithète

- Les attributs

  Les adjectifs en position d'attribut ancrent des *AES* comprenant les nœuds des verbes copulatifs et de leur sujet. Il est ainsi possible d'imposer des contraintes sur le nom dont ils peuvent être attributs.

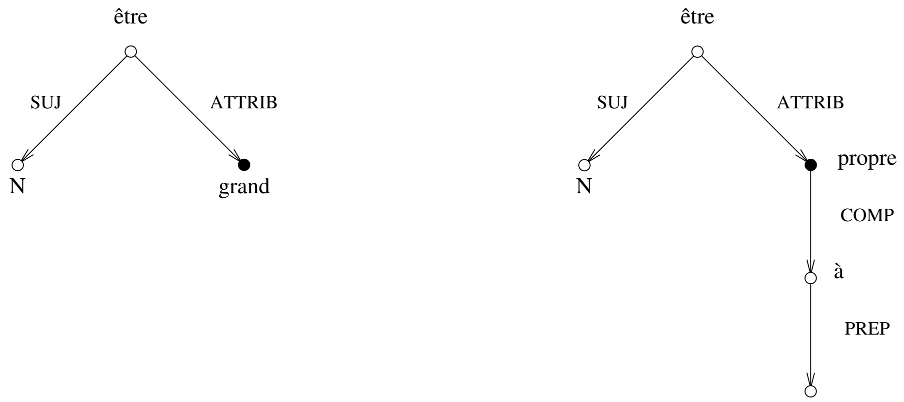


FIG. B.12: Adjectifs en position d'attribut

B.1.6 Les adverbes

Les adverbes ancrent des *AES* qui portent à leur racine la catégorie qu'ils peuvent modifier, verbe ou adjectif. S'il n'y a pas moyen de distinguer structurellement les adverbes modifiant un verbe ou une phrase, il est par contre possible de les distinguer par l'intermédiaire de l'étiquette de la dépendance liant le verbe et l'adverbe. On peut prévoir deux dépendances différentes *adv* et *adv-prop* permettant de distinguer les deux types d'adverbes.

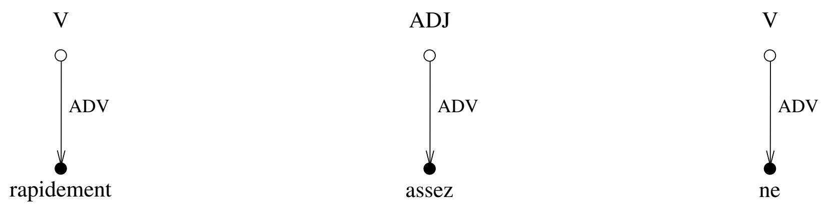


FIG. B.13: Adverbes

B.1.7 Les pronoms relatifs sujet et objet

Nous avons associé aux pronoms relatifs sujet et objet des *AES* complexes dans lesquels sont représentés le verbe de la relative et l'antécédent du pronom relatif. Il est important de remarquer qu'il n'existe pas de dépendance syntaxique entre le nom et le pronom. Cette représentation des pronoms relatifs permet d'utiliser un *AES* verbal standard dans une proposition relative, du fait que la dépendance liant le nom au verbe de la relative est représentée dans l'*AES* du pronom relatif.

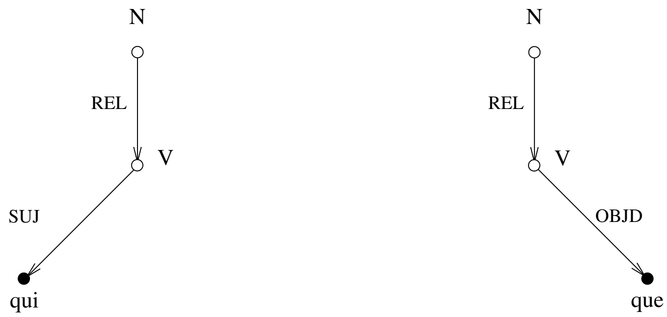


FIG. B.14: Pronoms relatifs sujet et objet

B.1.8 Les conjonctions de coordination

La coordination possède en syntaxe de dépendance un statut controversé. Certains auteurs, tels que [Tesnière, 1959] ou [Hudson, 1984] considèrent que la coordination ne peut être décrite à l'aide de dépendances syntaxiques. De nouveaux moyens de représentation sont alors introduits pour représenter la coordination, tel que la jonction chez L. Tesnière. D'autres auteurs, dont I. Mel'čuk représentent la coordination à l'aide de dépendances. Dans ce dernier cas, deux types de structures peuvent être utilisées, des structures symétriques, dont la tête est constituée de la conjonction de coordination ou des structures assymétriques dans lesquelles c'est un des membres de la coordination qui constitue la tête. I. Mel'čuk a opté pour la deuxième construction, arguant que les membres d'une structure coordonnée, n'ont souvent pas le même statut (*Il s'est levé et m'a tendu la lettre* $\neq$ *Il m'a tendu la lettre et s'est levé*). Pour notre part, nous avons choisi la structure symétrique, pour des raisons pratiques, d'accord entre la structure coordonnée et son gouverneur (*Paul et Marie* **sont venus**). Nous avons attribué à la conjonction de coordination la catégorie des membres de la coordination et avons ajouté un trait *coord = 1* indiquant qu'il s'agit d'une structure coordonnée, permettant ainsi de prendre en compte naturellement la nature endocentrique de la coordination. Différents *AES* associés à la conjonction de coordination, selon la catégorie des membres de la coordination ont été représentés dans la figure B.15.

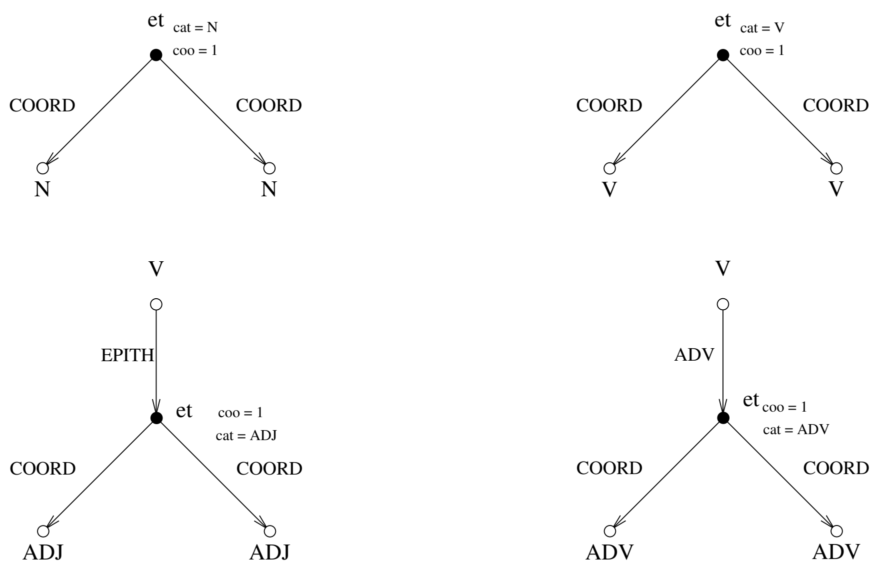


Fig. B.15: Conjonctions de coordination

B.2 Les règles de paraphrase

Les règles de paraphrase telles qu'elles sont définies dans la *TST* et qui sont représentées dans notre modèle par une relation entre $\mathcal{AEP}$ ne représentent que les paraphrases lexicales où les deux phrases paraphrastiques se distinguent obligatoirement par au moins une unité lexicale pleine. Les paraphrases syntaxiques, issues de choix syntaxiques différentes, sans modifications lexicales, tel que le changement de diathèse ou la transformation d'une relative objet en participe passé, sont représentées par les règles d'abstraction reliant un $\mathcal{AEP}$ à un $\mathcal{AES}$ ou par des règles de linéarisation. Nous allons décrire ici les quelques règles de paraphrase empruntées à [Mel'čuk, 1988b].

B.2.1 Substitutions synonymiques

Les substitutions synonymiques représentent les règles de paraphrase les plus simples. Elles se bornent généralement à remplacer un lexème par un autre, les deux étant liés par la fonction lexicale *Syn*.

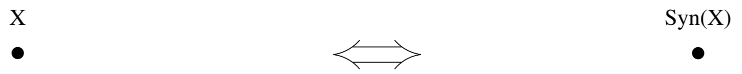


FIG. B.16: Règle de substitution synonymique

Exemple : *Jean a acheté une automobile* $\longrightarrow$ *Jean a acheté une voiture*

B.2.2 Substitutions conversives

Les substitutions conversives permettent de remplacer deux lexèmes conversif, tels que *acheter* et *vendre*. La règle effectue le remplacement tout en modifiant les fonctions syntaxiques profondes des actants. Les substitutions conversives sont basées sur des fonctions lexicales conversives. Plusieurs fonctions lexicales conversives sont distinguées selon le nombre d'actants possédés par les lexèmes liés et les permutations. La fonction $Conv_{321}$, par exemple, permet de lier deux lexèmes trivalents et tels que le troisième actant de l'un permute avec le premier actant de l'autre. Nous avons représenté dans la figure B.17 la règle conversive associée à la fonction lexicale $Conv_{321}$.

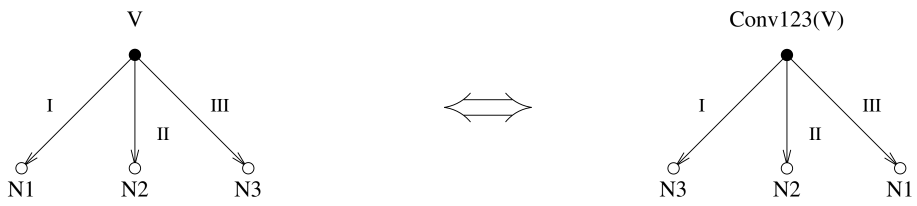


FIG. B.17: Une règle de paraphrase conversive

Exemple :
Jean a donné un livre à Marie $\longrightarrow$*Marie a reçu ce livre de Jean*

B.2.3 Fission à négation

La fission à négation permet de substituer à un verbe ou à un attribut un autre verbe ou attribut antonyme auxquels ils sont liés par la fonction lexicale *Anti*. Le remplacement nécessite l'introduction d'un adverbe de négation. Nous avons représenté dans la figure B.18 la règle de remplacement d'un attribut par un antonyme.

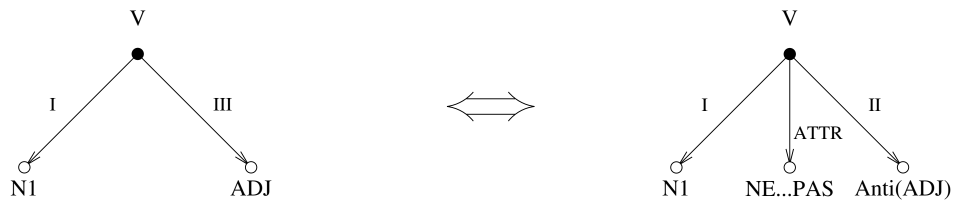


FIG. B.18: Règle de fission à négation

Exemple :
Ce bateau est grand $\longrightarrow$*Ce bateau n'est pas petit*

B.2.4 Fission à verbe support

Il existe trois familles de fonctions lexicales ($Oper_i$, $Func_i$ *et* $Labor_{ij}$) liant un nom N à ses verbes support. Lorsque $V = Oper_i(N)$, N est sujet de V et l'actant profond numéro i de N est objet direct de V. Lorsque $V = Func_i(N)$, N est objet direct de V et l'actant profond numéro i de N est sujet de V. Enfin, lorsque $V = Labor_{ij}(N)$, l'actant profond numéro i de N est sujet de V et l'actant profond numéro j de N est l'objet direct de V. A chaque fonction lexicale correspond une règle permettant de remplacer un verbe sémantiquement plein par une construction à verbe support. Nous avons représenté, dans la figure B.19 la règle correspondant à la fonction $Oper_2$.

Exemple :
Jean conseille Marie $\longrightarrow$*Jean donne un conseil à Marie*

B.2.5 Fission à copule

La règle de fission à copule permet de transformer une structure de type attribut de l'objet en structures verbales à complétive objet ayant *être* pour verbe. La fonction lexicale liant le verbe copule au verbe de la proposition principale est $Conv_{12-est-3}$. La règle est représentée dans la figure B.20.

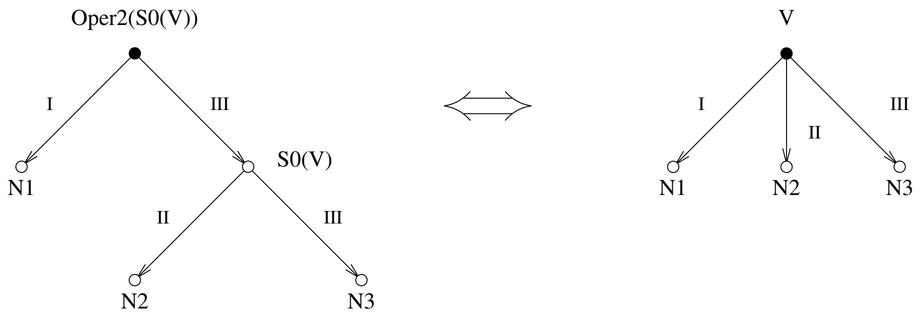


FIG. B.19: Règle de fission à verbe support

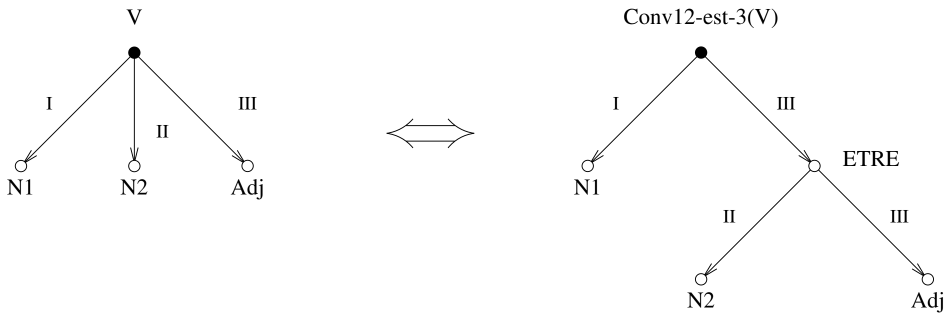


FIG. B.20: Règle de fission à copule

Exemples :
Je sais Jean malade ⟶*Je sais que Jean est malade*
Posons le diamètre du cercle égal à dix ⟶*supposons que le diamètre du cercle est égal à dix*

B.2.6 Substitution de verbes support

Certaines règles permettent de substituer un verbe support à un autre. Il s'agit de règles particulières dans la mesure où elles mettent en jeu des lexèmes (les deux verbes supports) qui ne sont pas liés directement par une fonction lexicale. Il existe autant de règles que de combinaisons de types de verbes support. Nous avons représenté dans la figure B.21 la règle permettant la substitution des verbes supports $Oper_1$ et $Func_1$.

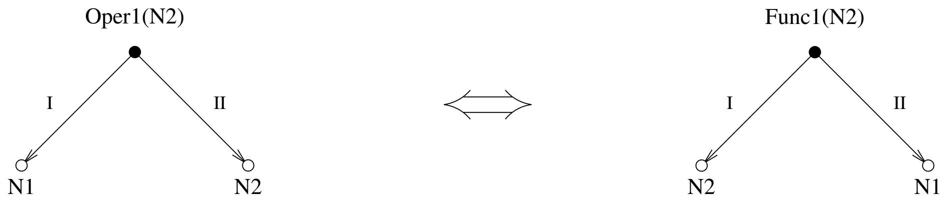


Fig. B.21: Une règle de substitution de verbes supports

Exemple :
Jean a posé cette question ⟶*Cette question est venue de Jean*

Annexe C

Eléments d'implémentation

Nous allons décrire dans cette annexe certains aspects de l'implémentation en LISP (Le-Lisp 15.260) de l'analyseur syntaxique. Ce dernier correspond à la partie la plus complexe du processus de reformulation et c'est sur lui qu'a porté l'essentiel de l'effort d'implémentation. Nous commencerons par décrire l'implémentation des structures de données, puis celle des traitements les plus importants.

C.1 Structures de données

C.1.1 Données implémentées sous la forme de structures de traits complexes

Nous allons décrire dans cette section les données représentées sous la forme de Structures de Traits Complexes (STC). Nous commencerons par décrire la façon dont nous avons implémenté les STC en LISP. Nous décrirons ensuite les données représentées sous la forme de STC, à savoir les lexèmes, les nœuds d'arbres syntaxiques et les arbres syntaxiques.

Implémentation des structures de traits complexes

Nous avons représenté dans la figure C.1 une STC représentée sous la forme d'un graphe acyclique orienté étiqueté. On peut distinguer dans le graphe trois types d'entités, les arcs (a1, a2, a3, a4), les nœuds non terminaux (n1, n3) et les nœuds terminaux (n2, n4, n5).

Ces trois entités sont implémentées différemment. Les nœuds terminaux sont représentés sous forme de chaînes de caractères ou de numéraux dont la valeur est égale à l'étiquette du nœud terminal. Les nœuds non terminaux sont implémentés par des symboles, le p-name du symbole étant l'étiquette du nœud. Les arcs émanant d'un nœud sont représentés dans la p-list (liste de propriété) du symbole correspondant à ce nœud. La p-list d'un nœud non terminal est une liste de la forme

(arc1 noeud1 arc2 noeud2 arc3 noeud3 ...)

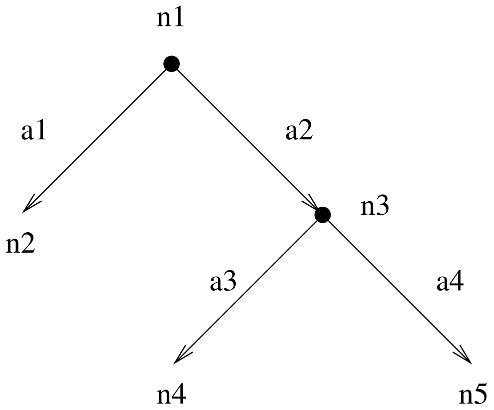


Fig. C.1: Représentation graphique d'une STC

représentant les arcs sortant du symbole et, pour chaque arc, le nœud constituant son extrémité.

Afin de nous rapprocher de la terminologie employée habituellement dans les STC, nous parlerons d'*attribut* au lieu d'étiquette d'arc. L'étiquette des nœuds situés à l'extrémité de chaque arc constituera la *valeur* de l'attribut représenté par l'arc. Une paire (attribut, valeur) sera appelée un trait. Lorsqu'un attribut a pour valeur un nœud non terminal, on parlera d'attribut complexe, de valeur complexe et de trait complexe. Lorsque la valeur de l'attribut est un nœud terminal, on parlera d'attribut simple, de valeur simple et de trait simple. La p-list se présente donc comme une liste alternée d'attributs et de valeurs ou comme une liste de traits. Les p-list des nœuds non terminaux de la figure C.1 sont représentés ci-dessous :

```
?  (plist 'n1)
= (a1 n2 a2 n3)
?  (plist 'n3)
= (a3 n4 a4 n5)
```

La représentation des nœuds non terminaux des STC sous forme de symboles liés entre eux par l'intermédiaire de leur p-list nous a semblé intéressante pour le parcours des STC lors de l'opération d'unification et du traitement de la pile graphe.Une représentation plus classique, sous forme de listes imbriquées, aurait abouti à la création de listes très longues et de procédures de parcours plus coûteuses.

La représentation des nœuds non terminaux par des symboles est par contre coûteuse en encombrement mémoire car à un symbole sont associées six propriétés dont seules trois sont utilisées ici (le p-name, la p-list et la objval (voir ci-dessous)). C'est en raison du coût des symboles que nous avons utilisé des chaînes de caractères et des numéraux pour représenter les nœuds terminaux. Ces derniers n'ayant pas de fils, ils n'ont pas besoin d'une p-list.

Quelques enrichissements

- Attributs répétables

  Les STC ne permettent pas de représenter au niveau d'une valeur complexe plusieurs attributs de même nom, ce qui revient, dans la représentation arborescente, à définir au niveau d'un nœud non terminal donné plusieurs arcs portant la même étiquette. Cette limite pose des problèmes, en grammaire de dépendances, pour représenter les dépendances répétables. Pour pallier à cet inconvénient, nous allons enrichir les STC en introduisant des attributs répétables. Ils permettent de représenter une structure dans laquelle plusieurs nœuds non terminaux sont reliés par un arc de même nom à un même père. Ce sont des attributs répétables qui sont utilisés pour représenter les dépendances répétables des arbres syntaxiques. La valeur d'un attribut répétable est représentée dans une `p-list` par une liste de symboles, la liste des nœuds liés à un même père le sont par des dépendances de même étiquette.

- Attributs à valeur de liste

  Certains attributs peuvent avoir pour valeur une liste de valeurs simples. Nous parlerons dans ce cas d'attributs à valeur de liste. Il est important de noter que les listes ne peuvent comporter que des valeurs simples. On trouvera comme exemple d'attributs à valeur de liste, l'attribut `sep` défini au niveau des nœuds d'arbres syntaxiques. La valeur d'un attribut à valeur de liste est représentée dans une `p-list` par une liste de chaînes de caractères ou de numéraux.

- Disjonctions de valeurs simples

  Certains attributs peuvent avoir pour valeur une disjonction de valeurs simples. Une disjonction peut être utilisée en place de toute valeur simple. Elle est utilisée en particulier pour l'attribut catégorie (`cat`), au niveau des lexèmes. La disjonction de valeurs complexes n'a pas été implémentée car elle peut provoquer une explosion exponentielle ([Kasper, 1987]). Une disjonction des trois valeurs simples `a, b` et `c` est représentée par l'expression `($a b c)`.

- Traits non soumis à l'unification

  Les informations représentées sous forme de STC seront soumises à l'unification au cours des traitements. Nous avons toutefois prévu une manière de représenter des traits qui échapperont à l'unification : ces traits sont représentés dans la `objval` des symboles. Cette dernière se présente comme une `p-list` (liste alternée d'attributs et de valeurs). On trouvera de tels traits au niveau des nœuds d'arbres syntaxiques (le trait `pere`). Lors de l'unification de deux STC comportant des traits non soumis à l'unification, un traitement particulier sera réservé à ces derniers.

Une STC enrichie des traits inroduits ci-dessus est représentée dans la figure C.2. Les attributs non soumis à l'unification ont été représentés par des arcs en pointillés, les valeurs simples sont identifiées par des expressions de la forme *vN* et les valeurs complexes par des expressions de la forme *vcN*.

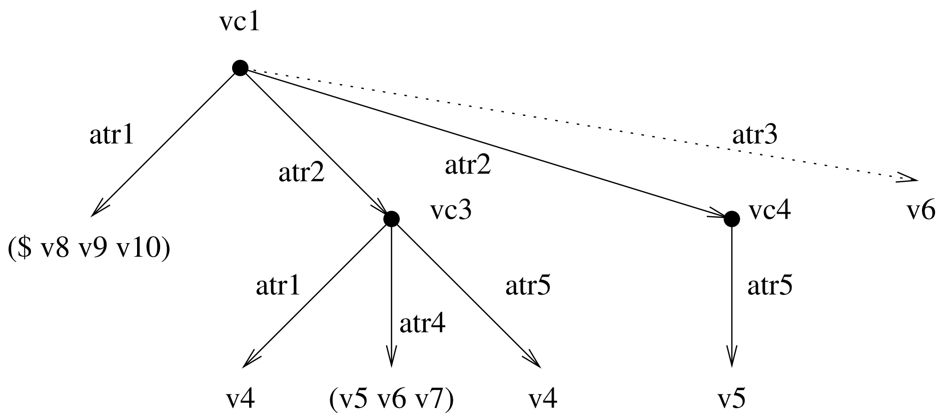


FIG. C.2: Représentation graphique d'une STC enrichie

Les lexèmes

Un lexème est implémenté par un symbole dans la `p-list` duquel sont représentés ses différents traits. On retrouve dans la `p-list` tout genre de traits morphologiques et sémantiques utiles, plus un attribut `pos` ayant pour valeur la position du lexème dans la phrase, lorsque cette information est pertinente. Le `p-name` du symbole est de la forme `lexN`, où `N` est une valeur numérique. Le lexème correspondant au verbe *mangeaient*, situé, par exemple, en troisième position[1] dans la phrase, aura pour `p-list` la liste suivante :

```
(lex manger-1 cat v tps impar mod ind pers 3 num 2 pos 2)
```

Un tel lexème est représenté sous forme graphique dans la figure C.3.

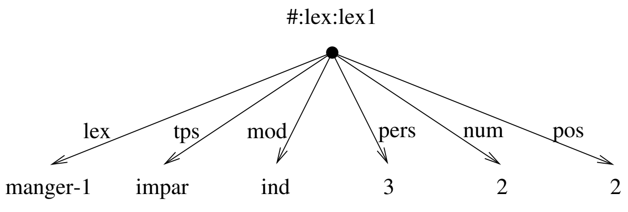


FIG. C.3: Représentation graphique d'un lexème

Nous ne donnerons pas la liste complète des attributs possibles d'un lexème, dans laquelle on

[1] Le premier mot de la phrase est en position zero

pourra trouver, en plus des traits `lex` et `pos`, les attributs `cat num gnr tps mod per cas`. Ces traits dépendent du dictionnaire ou de l'analyseur morphologique utilisé.

Les nœuds des arbres syntaxiques

Les nœuds sont aussi représentés par des symboles. Les conventions de nommage du nœud (valeur de leur `p-name`) sont différentes selon que le nœud est un nœud d'*AES* ou un nœud d'arbre obtenu par unification. Dans le premier cas, le `p-name` du nœud est de la forme `aesN@X` où `aesN` est le nom de l'*AES* auquel appartient le nœud (voir les conventions de nommage des *AES* ci-dessous) et `X` est l'adresse du nœud. Ainsi, on peut directement accéder à un nœud d'*AES* en connaissant le nom de l'*AES* et l'adresse du nœud. Dans le second cas, le `p-name` est de la forme `nN`, où `N` est un entier.

Un nœud N comporte dans sa `p-list` les attributs suivants :

- `lexeme`, attribut complexe ayant pour valeur le lexème associé à N.
- `ancre`, attribut simple ayant pour valeur 1 s'il s'agit d'un nœud noir et () sinon.
- `cote`, attribut simple ayant pour valeur le caractère `d` si N est situé à droite de son gouverneur et `g` s'il est situé à sa gauche.
- `sep`, attribut à valeur de liste ayant pour valeur la liste des frères potentiels de N pouvant le séparer de son gouverneur. Les frères potentiels sont représentés par leur fonction (l'étiquette de l'arc les reliant à leur père).
- `oblig`, attribut simple ayant pour valeur 1 si le nœud est obligatoire : s'il doit nécessairement être identifié à un mot de la phrase analysée.
- `naes`, attribut à valeur de liste dans laquelle sont représentés, sous forme de valeurs simples (voir C.1.4), les nœuds d'arbres élémentaires ayant été unifiés entre eux pour constituer N. C'est grâce à ce trait que pourra être construite la structure syntaxique de surface analytique. On trouvera plus de détails sur ce trait ci-dessous, lors de la description de la représentation linéaire des arbres.
- `pere`, attribut complexe non soumis à l'unification qui a pour valeur le nœud père de N (son gouverneur). Le trait `pere` n'est pas représenté comme un trait complexe normal car il introduit des cycles dans les arbres syntaxiques (un dépendant est relié à son gouverneur qui, à son tour, est relié au dépendant ...).
- Attributs complexes de la forme `&dep` où `dep` est un nom de dépendance (par exemple : `&sujet`, `&agent`). Ils ont pour valeur un dépendant de N.
- Attributs répétables de la forme `£dep` où `dep` est un nom de dépendance répétable (par exemple : `£epithete`, `£circons`). Ils permettent de réprésenter des dépendances répétables et ont pour valeur une liste de nœuds : les dépendants de N liés à ce dernier par des dépendances de même étiquette.

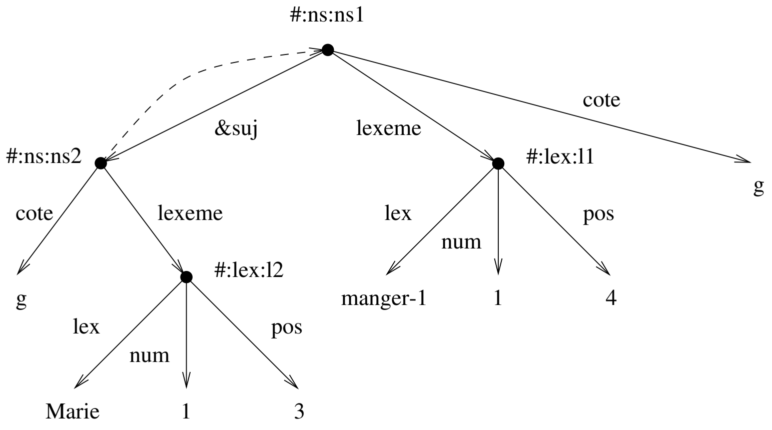


FIG. C.4: Représentation graphique de nœuds d'arbre syntaxique liés entre eux

Un exemple de nœuds d'arbres liés entre eux est représenté dans la figure C.4.
Nous avons repris ci-dessous la p-list du nœud `ns1` de la figure C.4.

```
?  (plist 'ns1)
= (lexeme l1 cote g &suj ns2)
```

Les arbres syntaxiques

Un arbre syntaxique est implémenté par un symbole dont le p-name constitue le nom de l'arbre syntaxique. Lorsqu'il s'agit d'un arbre élémentaire, son nom est de la forme `aesN`. Sinon, son nom est de la forme `aN`. Un arbre comporte trois attributs :

- `rac`, attribut complexe ayant pour valeur un nœud syntaxique, la racine de l'arbre.
- `gauche`, attribut complexe ayant pour valeur le nœud noir de l'arbre correspondant au mot de la phrase situé le plus à gauche.
- `droite`, attribut complexe ayant pour valeur le nœud noir de l'arbre correspondant au mot de la phrase situé le plus à droite. Les deux traits `droite` et `gauche` permettent d'implémenter la notion d'*extension* d'un nœud X (le segment de phrase correspondant au sous-arbre dont le nœud X constitue la racine).
- `type`, attribut simple, n'apparaîssant que sur les arbres élémentaires. Il représente le type de l'arbre élémentaire.

Un exemple d'arbre non élémentaire est représenté sous forme graphique dans la figure C.5

Un exemple de p-list d'un arbre élémentaire est représentée ci-dessous :

```
? (plist 'aes1)
= (droite aes1@1 gauche aes1@1 type art1*n rac aes1@0)
```

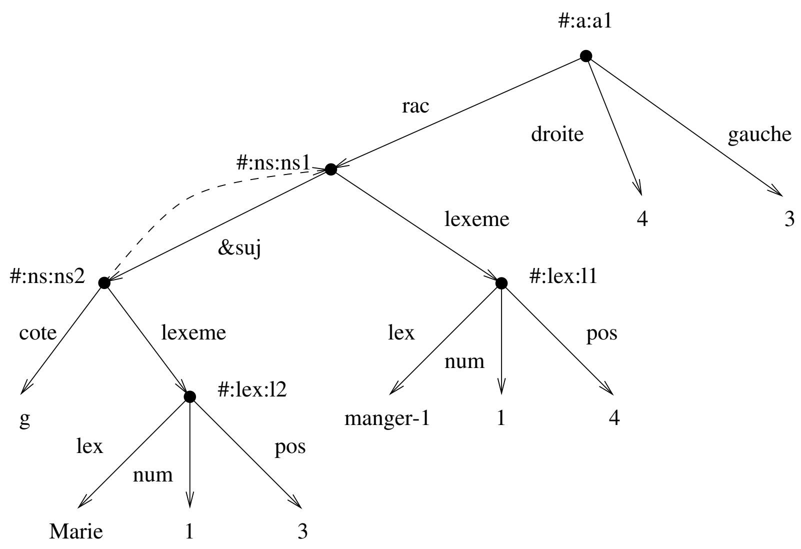


FIG. C.5: Représentation graphique d'un arbre syntaxique

C.1.2 Autres données représentées sous forme de graphes

D'autres données sont représentées sous forme de graphe sans toutefois constituer des STC, du fait qu'elles peuvent comporter des cycles et qu'elles ne sont pas soumises à l'unification. Elles sont par contre implémentées suivant le même principe que les STC : des symboles liés entre eux par l'intermédiaire de leurs `p-list`.

La pile graphe

Une pile graphe est un graphe étiqueté orienté pouvant comporter des cycles. Une pile graphe est constituée de cellules, chacune étant représentée par un symbole. Ce dernier a pour `p-name` `cN`, où `N` est un nombre. Une cellule comporte trois attributs :

- `l-prec` attribut répétable ayant pour valeur la liste des cellules précédentes.
- `l-suiv` attribut répétable ayant pour valeur la liste des cellules suivantes.
- `arbre` attribut complexe ayant pour valeur l'arbre syntaxique associé à la cellule. La cellule origine de la pile graphe ne possède pas d'attribut `arbre` car il ne lui correspond pas d'arbre.
- `l-ext` attribut répétable ayant pour valeur la liste des cellules formant l'extrémité de la pile graphe (les cellules n'ayant pas de successeurs). Il n'est défini que pour la cellule origine de la pile graphe. Sa présence n'est pas obligatoire, les extrémités pouvant être décelées grâce à un parcours, mais elle permet d'accélérer les traitements.

Une pile graphe a été représentée sous forme graphique dans la figure C.6. Une représentation simplifiée, qui sera utilisée dans la suite de ce document apparaît dans la figure C.7.

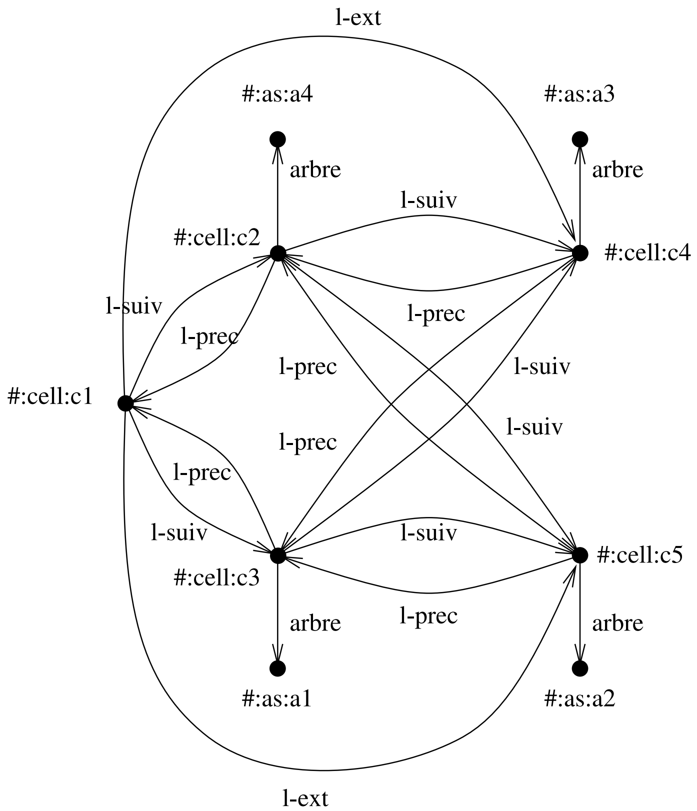


Fig. C.6: Représentation graphique d'une pile graphe

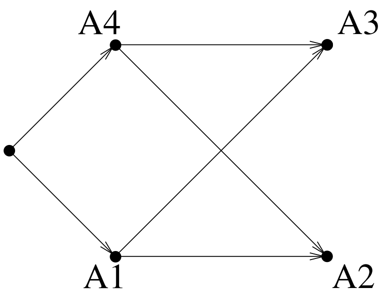


Fig. C.7: Représentation graphique simplifiée d'une pile graphe

Le dictionnaire de mots composés

Le dictionnaire des mots composés permet de représenter les suites de lexèmes formant un mot composé (cahier-1 de-1 le-1 charge-1 → cahier-des-charges). Il se présente sous une forme arborescente dont les arcs sont étiquetés par des lexèmes. Les nœuds sont représentés par des symboles de `p-name` `mcN` où `N` est la valeur du compteur `ctr-mn` au moment de la création du nœud. Le chemin menant de la racine à un nœud quelconque correspond à une suite de lexème constituant, dans certains cas, un mot composé. La structure arborescente du dictionnaire permet de regrouper tous les mots composés commençant par une même suite de lexèmes et d'optimiser le parcours du dictionnaire. Ainsi, les deux mots composés correspondant aux suites de lexèmes *clapet-1 de-1 surpression-1* et *clapet-1 de-1 sectionnement-1* verront leur deux premiers éléments représentés une seule fois dans le dictionnaire de mots composés (voir figure C.8). Les nœuds de la structure comportent les attributs suivants :

- Attributs correspondants à des arcs. Ils ont pour nom le lexème qui étiquette l'arc. Il est possible d'imposer à un arc des conditions sur la valeur de certains traits morphologiques. On peut ainsi imposer, dans le mot composé *cahier-des-charges*, que le substantif *charge* et son déterminant soient au pluriel. L'arc correspondant au lexème *charge-1* dans chemin correspondant au mot composé *cahier des charges* se présente de la façon suivante :

  ```
  (charges-1 ((num 2) mc12))
  ```

  Il indique que l'arc étiqueté par le lexème *charge-1* est associé à la condition *num = 2*. Plus généralement, un arc se représente de la façon suivante : `(lexème ( liste-de-conditions noeud))`.

- `@lex` attribut à valeur de liste, il n'est défini que pour les nœuds correspondant à des extrémités de chemins correspondant à un mot composé. La valeur de l'attribut consiste en une liste de la forme :

  ```
  (N (liste-de-traits) mot-composé)
  ```

  décrivant le mot composé correspondant au chemin où `N` est un nombre indiquant le lexème du mot composé dont le mot composé hérite des traits morphologiques et lexicaux. Dans le mot composé correspondant à la suite *cahier de les charges*, par exemple, le mot composé héritera des traits du premier mot : *cahier*. Lorsque le mot composé possède des trait n'apparaissant dans aucun des mots le composant, ces traits peuvent être introduits au niveau de la `liste-de-traits`. Ces traits s'ajouteront ou écraseront les traits du mot dont le mot composé hérite.

Nous avons représenté dans la figure C.8 un fragment de dictionnaire de mots composés. On notera la factorisation des deux mots composés *clapet de surpression* et *clapet de sectionnement*, les conditions associés aux lexèmes *le-1* et *charge-1* dans le mot composé *cahier des charges*. On notera, enfin, le trait *cat=conjcoo* au niveau du mot composé *ainsi-que*, indiquant que ce dernier est de la catégorie conjonction de coordination alors qu'aucun de ses composant ne l'est.

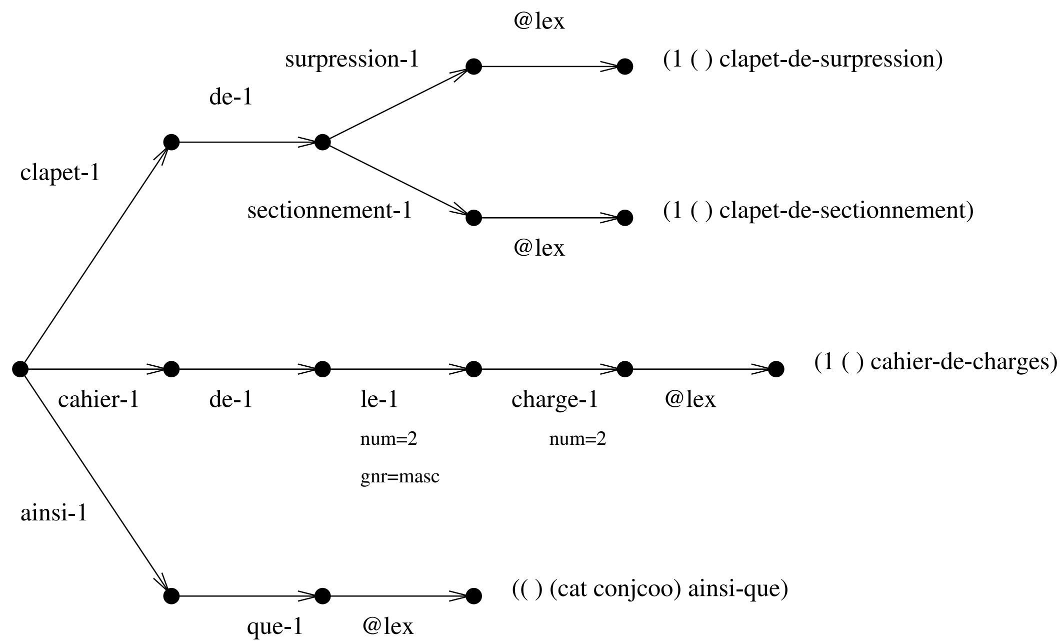


FIG. C.8: Représentation graphique d'un fragment de dictionnaire de mots composés

C.1.3 Données représentées sous la forme de tables

Le dictionnaire morphologique

Le dictionnaire morphologique est implémenté par une table de hachage ayant pour clef un symbole dont le `p-name` correspond à une forme fléchie. Une entrée se décompose en plusieurs sous-entrées, chacune correspondant à une combinaison de traits morphologiques. L'entrée de la forme fléchie *volais*, par exemple, se décompose en deux sous-entrées, l'une correspondant au verbe *voler* à la première personne de l'imparfait de l'indicatif et l'autre reservée au verbe *voler* à la seconde personne de l'imparfait. De plus, une sous-entrée peut correspondre à plusieurs lexèmes. Chaque sous-entrée de *volais*, par exemple, correspond aux deux lexèmes *voler-1* et *voler-2*, le premier étant un synonyme de *dérober* et le second, synonyme de *évoluer dans les airs*.

L'entrée correspondant à *volais*, décrite ci-dessus, se présente de la façon suivante :

```
((lex ("voler-1" "voler-2") tps "imparfait" mod "ind" pers 1 num 1)
 (lex ("voler-1" "voler-2") tps "imparfait" mod "ind" pers 2 num 1))
```

Pour simplifier l'écriture du dictionnaire, lorsqu'une sous-entrée correspond à un seul lexème, la valeur de l'attribut `lex` n'est pas représentée dans une liste :

```
((lex "voler-1" tps "imparfait" mod "ind" pers 1 num 1))
```

Le dictionnaire lexical

Le dictionnaire lexical est implémenté par une table de hachage ayant pour clef la chaîne de caractères correspondant à un lexème. L'entrée correspondant à un lexème est constituée de la liste des différents $\mathcal{AES}$ (types d'*AES*) qu'il peut ancrer ainsi que des fonctions lexicales qui le lient à d'autre lexèmes. La liste des $\mathcal{AES}$ que le lexème peut ancrer se décompose en sous-listes auxquelles sont associées des conditions portant sur la valeur de certains traits morphologiques. En effet, plusieurs formes fléchies, correspondant à un même lexème, peuvent ancrer des $\mathcal{AES}$ différents selon la valeur de leurs traits morphologiques. Les deux formes fléchies *volait* et *volé*, associés au même lexème *voler-1*, par exemple, ancreront des $\mathcal{AES}$ différents. Une entrée correspondant au lexème *voler-1* à été représentée ci-dessous.

```
(aes (((fin "fini") (n1v*n2))
      ((fin "ppe") (n1avoirppe2*n20 n1etreppe2*par20n200)))
Syn (dérober-1) S0 (vol-1))
```

Une telle entrée indique qu'une forme fléchie correspondant au lexème *voler-1* peut ancrer l'$\mathcal{AES}$ `n1v*n2`[2] si son attribut `fin` a pour valeur `fini` (indiquant que le verbe est conjugué à un temps fini). Lorsque le trait `fini` a pour valeur `ppe` (participe passé), le verbe peut ancrer les $\mathcal{AES}$ `n1avoirppe2*n20` et `n1etreppe2*par20n200`. Deux fonctions lexicales sont de plus associées au lexème *voler-1* : la fonction *Syn* le liant à son synonyme *dérober-1* et la fonction S_0 le liant à son dérivé syntaxique nominal *vol-1*.

C.1.4 Représentation linéaire des arbres syntaxiques

La représentation des arbres syntaxiques sous forme de STC a été décrite ci-dessus. Un autre mode de représentation, correspondant à la représentation analytique est utilisé pour stocker et comparer à faible coût des arbres syntaxiques. Il se présente sous forme d'une liste de listes de nœuds d'*AES*.

Nous avons représenté côte à côte, dans la figure C.9, un arbre A et la composition d'*AES* lui correspondant. Nous avons indiqué de plus, au niveau des nœuds de A, les nœuds d'*AES* dont

[2] Les noms de types d'arbres élémentaires suivent la convention suivante : chaque nœud de l'arbre est représenté par sa catégorie ou sa valeur lexicale, si cette dernière est précisée, suivie de son adresse au sein de l'arbre élémentaire. Les adresses des nœuds suivent aussi une convention : le nœud racine a toujours 0 pour adresse, ses dépendants ont pour adresse 1, 2, 3 ... Leurs dépendants ont respectivement pour adresse 10, 11, 12 ... 20, 21, 22 ... De plus, le nœud correspondant à l'ancre lexicale est suivi d'une astérisque. Les différents nœuds sont concaténés en suivant plus ou moins rigoureusement l'ordre grammatical standard. `n1v0*n2à3n30` correspond à un type d'arbre élémentaire à la racine duquel se trouve un verbe, ancre de l'arbre élémentaire (`v0*`). Le verbe possède trois dépendants : deux noms et la préposition *à* (`n1 n2 à3`). Cette dernière possède un dépendant nominal (`n30`).

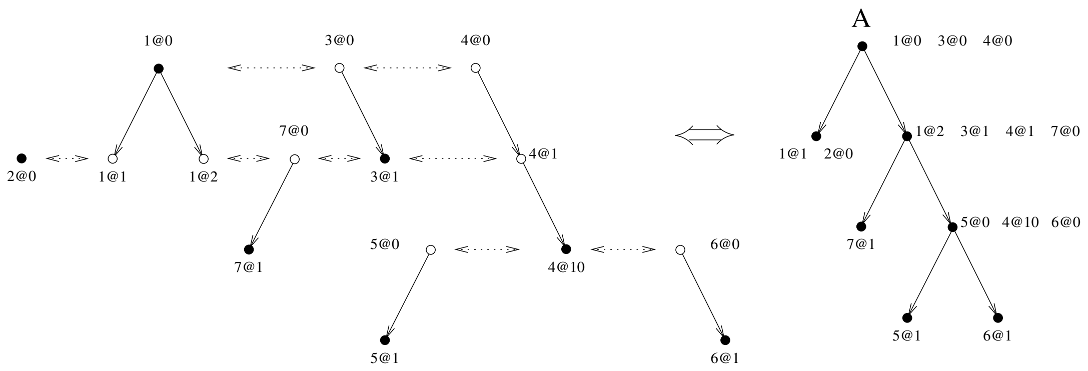


FIG. C.9: Décompostion d'un arbre en *AES*

il est issu[3]. Ces nœuds d'*AES* ont été stockés au fur et à mesure des opérations d'unification dans le trait `naes` des nœuds de A. Ce sont ces informations qui vont permettre de construire la structure analytique. Le principe consiste à regrouper, au sein d'une liste, les différents nœuds d'*AES* correspondant à chaque nœud de l'arbre. La structure linéaire correspondant à l'arbre de la figure C.9 s'écrit donc :

```
((1@0 3@0 4@0) (2@0 1@1) (1@2 7@0 3@1 4@1) (5@0 4@10 6@0))
```

On pourra remarquer que cette liste est redondante. Il n'est en particulier pas nécessaire de représenter le fait que `3@1` et `4@1` doivent être unifiés dans la mesure où il est déjà dit que leurs gouverneurs respectifs (`3@0` et `4@0`) doivent l'être. C'est pourquoi, au lieu de représenter dans chaque sous-liste l'ensemble des nœuds d'*AES* devant être unifiés, nous ne représenterons que les nœuds d'*AES* correspondant aux racines (de la forme `X@0`) et un unique nœud d'*AES* ne correspondant pas à une racine. Son rôle est d'assurer la cohésion de l'ensemble résultant des différentes unifications, la racine de chaque *AES* s'unifiant avec un nœud d'un autre *AES*. La représentation linéaire n'indique donc que les unifications des racines des arbres élémentaires avec un nœud non racine d'un autre arbre élémentaire. Ainsi, dans la sous-liste `(1@2 7@0 3@1 4@1)`, ne seront conservés que la racine `7@0` et un site de rattachement. Lorsque plusieurs sites existent dans la liste, nous garderons celui dont le nombre précédant le caractère `@` est le plus petit, ici `1@2`. Dans le cas particulier de la racine de l'arbre, tous les nœuds d'*AES* unifiés correspondent à des racines. Dans ce cas, la sous-liste correspondant au nœud est constituée exclusivement de racines d'*AES*. Afin de permettre une comparaison rapide des structures linéaires, celles-ci sont triées à un double niveau. Chaque sous-liste, à l'exception de la sous-liste racine débutera par le nœud non racine que suivront les différentes racines triées par ordre croissant.

Exemple :

`(5@0 2@0 3@2 4@0)` $\rightarrow$ `(3@2 2@0 4@0 5@0)`

[3]Les nœuds d'*AES* sont représentés dans l'écriture linéaire des arbres sous une forme compacte, dans laquelle `X@Y` correspond au nœud `aesX@Y`

De plus, les sous-listes sont triées au sein de la liste complète. On trouvera en tête de liste la sous-liste correspondant à la racine, suivie des autres sous-listes triées par ordre croissant de leur premier élément.

Exemple :
`((2@1 4@0 5@0) (1@0 2@3 3@0) (1@2 9@0))` → `((1@0 2@3 3@0) (1@2 9@0) (2@1 4@0 5@0))`

Lorsque deux sous-listes débutent par les mêmes éléments, elles sont triées selon le premier élément qui les distingue.

Exemple :
`(3@2 4@0 6@0) (3@2 4@0 5@0 7@0)` → `(3@2 4@0 5@0 7@0) (3@2 4@0 6@0)`

La structure linéaire correspondant à l'arbre de la figure C.9 se présente donc de la façon suivante :

```
((1@0 3@0 4@0) (1@1 2@0) (1@2 7@0) (4@10 5@0 6@0))
```

C.1.5 Le tableau des résultats de l'analyse

Les résultats de l'analyse d'une phrase sont regroupés au sein d'un tableau N*N où N est la longueur de la phrase. Dans chaque case (G , D) ($0 \leq G \leq N-1$ $0 \leq D \leq N-1$) du tableau sont stockés, sous forme linéaire, les arbres correspondants au segment de phrase s'étendant du mot numéro G au mot numéro D. Le tableau est associé à la variable globale `tableau`.

C.2 Traitements

C.2.1 L'unification

Nous ne décrirons pas ici les principes de l'unification de structures de traits. On dira simplement que l'unification de deux STC A et B consiste à construire une nouvelle STC C qui comporte l'union des informations représentées par A et par B. Lorsque A et B sont contradictoires, attribuant des valeurs différentes aux mêmes attributs, l'unification échoue.

L'algorithme d'unification que nous avons implémenté est du type non destructif : à l'issue de l'unification de deux structures, ces dernières sont intactes. La raison du choix de ce mode d'unification est qu'une même structure peut participer successivement à plusieurs unifications. Ce mode d'unification est par contre coûteux en encombrement mémoire car une nouvelle structure est entièrement créée lors de chaque unification. De plus, lorsqu'une unification échoue, les structures qui ont été générées avant l'échec ne sont pas détruites.

On peut être amené, dans certains cas, à effectuer une opération d'unification dans le seul but de savoir si deux structures sont unifiables. C'est pourquoi deux opérations d'unification ont été implémentées : la première, appelée unification prédicative, vérifie que deux structures sont unifiables sans toutefois engendrer une nouvelle structure ; et la seconde crée une nouvelle structure. Les deux algorithmes suivent les mêmes principes généraux ; nous commencerons par la description du premier, qui est plus simple.

L'unification prédicative de deux STC consiste à les parcourir simultanément à partir de leurs racines dans un ordre de parcours du type *profondeur d'abord.* Deux pointeurs P_1 et P_2 sont positionnés sur les racines des deux STC. Lorsque deux arcs de même étiquette quittent les deux racines, les pointeurs sont positionnés sur les nœuds auxquels aboutissent les deux arcs. L'opération continue jusqu'à atteindre des nœuds terminaux. Les valeurs des nœuds terminaux sont alors comparées : si celles-ci sont égales, le parcours reprend à partir des derniers nœuds non terminaux possédant des arcs de même étiquette non encore parcourus ; si les valeurs sont différentes, l'unification échoue. L'unification aboutit lorsque le parcours s'achève sans échec ; cela signifie que les structures ne sont pas contradictoires.

Le principe du second algorithme est très proche du premier, en particulier ses conditions d'échec et de réussite sont les mêmes. La différence réside dans le fait qu'au cours du parcours, une nouvelle structure est créée. Ainsi, au départ, lorsque P_1 et P_2 sont positionnés sur les racines, un nouveau nœud est créé, constituant la racine de la nouvelle structure et un pointeur P_3 est positionné dessus. Dans la suite du parcours, lorsque les pointeurs P_1 et P_2 traversent un arc, un arc de même étiquette est créé dans la nouvelle structure, ainsi qu'un nœud sur lequel pointe maintenant P_3. Lorsque P_1 et P_2 aboutissent à deux nœuds terminaux et que ces derniers ont la même valeur, celle-ci est attribuée au nœud sur lequel pointe P_3. La seconde différence importante entre les deux algorithmes concerne les parties non communes de A et de B. Nous avons vu en effet que le parcours effectué lors de l'unification ne concerne que la partie commune de A et de B, car le succès ou l'échec de l'unification ne dépend que de cette partie. Dans le second alogorithme, les parties non communes de A et de B sont recopiées dans C. Ainsi, lorsque du nœud sur lequel pointe P_1 émane un arc n'ayant pas d'équivalent dans le nœud sur lequel pointe P_2, l'intégralité du sous-arbre situé à l'extrémité de la dépendance est recopiée et intégrée dans la nouvelle structure, au niveau du nœud sur lequel pointe P_3.

Les procédures décrites ci-dessus décrivent un processus d'unification classique. L'unification que nous avons implémentée s'en distingue un peu dans sa prise en compte des disjonctions, des attributs répétables, des attributs à valeur de liste et des attributs non soumis à l'unification.

- Dans les cas de disjonctions de valeurs simples (les seules autorisées) deux cas doivent être pris en compte : l'unification de deux disjonctions et l'unification d'une disjonction et d'une valeur simple. Dans le premier cas, l'intersection des deux disjonctions est calculée. Lorsque cette dernière est nulle, alors l'unification échoue. Dans le cas de l'unification d'une valeur simple et d'une disjonction, lorsque la valeur apparaît dans la disjonction alors l'unification réussit, sinon elle échoue.

  ```
  (Unif ($a b c) ($b c d)) → ($b c)
  (Unif ($a b c) ($d e f)) → ()
  ```

```
(Unif a ($a b c)) → a
(Unif a ($d e f)) → ()
```

- L'unification d'attributs répétables se fait par union des valeurs des deux attributs répétables. Ainsi, à l'issue de l'unification d'un nœud duquel émanent trois dépendances répétables étiquetées *Rep* et d'un nœud duquel émanent deux dépendances répétables d'étiquette *Rep*, sera créé un nœud duquel émanent cinq dépendances d'étiquette *Rep*. Il est important de remarquer que dans ce cas, il n'y a pas unification, à proprement parler, de dépendances, mais ajout de dépendance. C'est pour cette raison que l'unification de dépendances répétables ne peut échouer.

- L'unification d'attributs à valeur de liste est réalisée par union des deux listes.

  `(Unif (a b c) (c d e)) → (a b c d e))`

- L'attribut `pere` d'un nœud non terminal, liant le nœud à son nœud père ne peut être soumis à l'unification car il provoquerait une boucle sans fin. C'est pourquoi, lors de l'unification de deux nœuds non terminaux A et B, après la création d'un nouveau nœud C, le nœud issu de l'unification des pères respectifs de A et de B est donné comme valeur à l'attribut `pere` de C. L'attribut `pere` n'est pas nécessaire au bon fonctionnement de l'analyseur, mais sa présence permet d'accélérer certains traitements, principalement l'identification de sites d'attachement dans un arbre.

L'algorithme d'unification ne distingue pas les symboles associés aux lexèmes des symboles associés aux nœuds syntaxiques ou aux arbres syntaxiques, qui sont tous à ses yeux des nœuds non terminaux. C'est pourquoi, à l'issue d'une unification, on assistera à une certaine≪ uniformisation≫ des différents nœuds non terminaux : ils seront tous représentés par des symboles de `p-name nX`, où `X` est un nombre.

C.2.2 Manipulation de la pile graphe

Deux opérations sont définies au niveau de la pile graphe : l'opération d'empilement et l'opération de réduction.

L'empilement de N arbres dans la pile graphe se décompose en quatre étapes :

1. créer N nouvelles cellules.

2. associer un arbre à chaque cellule.

3. établir un lien entre chaque nouvelle cellule et les anciennes extrémités de la pile graphe.

4. Mettre à jour la liste des extrémités de la pile graphe.

Un exemple d'empilement est représenté dans la figure C.10. Trois arbres C, D et E sont empilés. Six liens sont établis entre les trois arbres et les deux extrémités précédentes de la pile graphe. A l'issue de l'empilement, C, D et E constituent les nouvelles extrémités de la pile graphe.

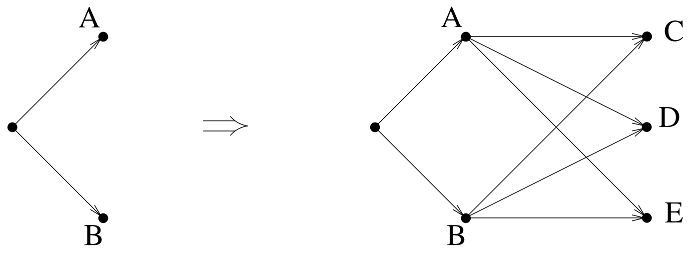


FIG. C.10: Empilement de trois arbres dans une pile graphe

L'opération de réduction de la pile graphe est plus complexe ; nous la décrirons à l'aide de deux opération plus simples : la réduction d'extrémité et la réduction élémentaire. La réduction d'une pile graphe consiste à réduire chacune de ses extrémités. La réduction d'une extrémité Ext revient à réaliser une réduction élémentaire des couples constitués de Ext et de chacune des cellules précédentes de Ext. La réduction élémentaire d'un couple de deux cellules A et B consiste à tenter l'attachement de l'arbre de A dans l'arbre de B et, inversement, l'attachement de l'arbre de B dans l'arbre de A. Lorsqu'une opération d'attachement aboutit à la création d'un ou plusieurs nouveaux arbres, alors le lien entre A et B est détruit, et pour chaque arbre, une nouvelle cellule est créée. Les cellules créées sont alors liées aux différents prédécesseurs de B. L'opération de réduction décrite ci-dessus est représentée dans la figure C.11. La pile graphe possède avant réduction deux extrémités A et C. La réduction de l'extrémité A consiste à effectuer la réduction élémentaire du couple A B et du couple A D. La réduction du couple A B consiste à effectuer l'attachement de l'arbre de A dans l'arbre de B et, inversement de l'arbre de B dans l'arbre de A. A l'issue de la réduction du couple A B, les nouveaux arbres $X_1 \ldots X_i$ sont créés, et le lien entre A et B est détruit.

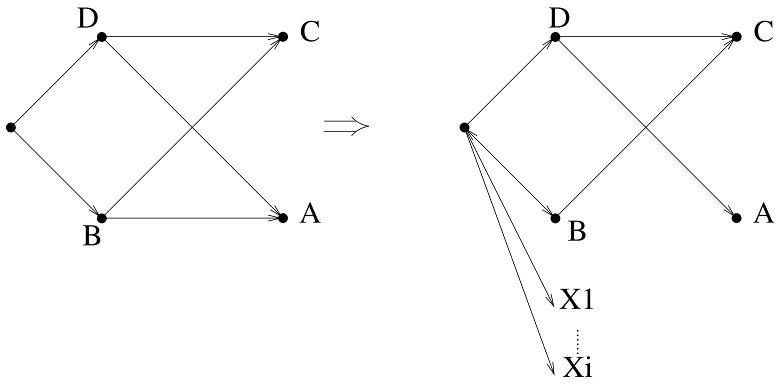


FIG. C.11: Réduction élémentaire du couple A B

Lorsque durant l'opération de réduction élémentaire de A et B une opération d'attachement au moins aboutit, les cellules A et/ou B sont détruites dans les deux cas suivants :

- Si B est l'unique prédecesseur de A, alors A est détruite, comme l'illustre la figure C.12.

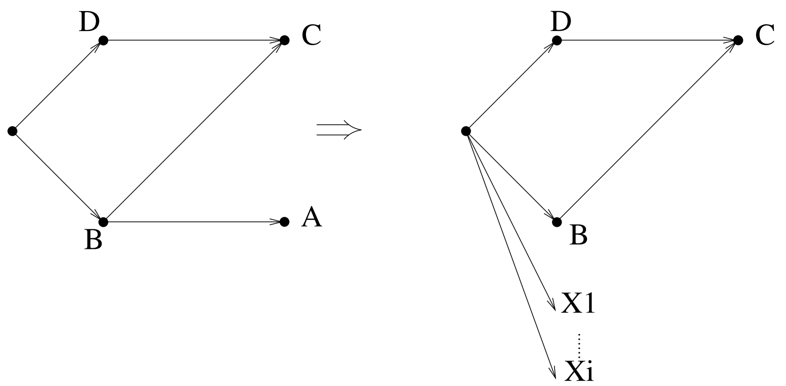


FIG. C.12: Destruction d'une cellule à l'issue d'une réduction

- Si A est l'unique successeur de B, alors B est détruite, comme l'illustre la figure C.13.

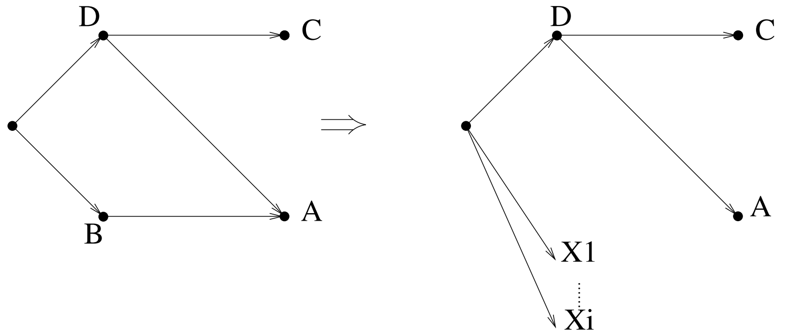


FIG. C.13: Destruction d'une cellule à l'issue d'une réduction

A l'issue de l'opération, la liste des extrémités de la pile graphe est mise à jour.

L'opération de réduction est récursive. Lorsqu'une opération d'attachement réussit et que de nouvelles cellules sont créées, constituant de nouvelles extrémités de la pile graphe, l'opération de réduction d'extrémité est relancée sur les nouvelles extrémitées créées.

Deux opérations de réduction, basées sur le schéma décrit ci-dessus, sont en fait définies : la réduction gauche et la réduction droite. La réduction droite est utilisée lorsque l'analyse est effectuée de gauche à droite. La réduction gauche est utilisée lorsque l'analyse est effectuée de droite à gauche. La différence entre les deux opérations réside dans les positions relatives des segments de phrase correspondant aux cellules sur lesquelles est effectuée la réduction. Considérons deux cellules C_{prec} et C_{suiv} de la pile graphe, telles que C_{prec} précède directement C_{suiv}.

Lorsque l'analyse est effectuée de gauche à droite, le segment de phrase correspondant à C_{suiv} est situé à droite du segment correspondant à C_{prec}. Ainsi, lors de la réduction, C_{suiv} est attachée dans C_{prec} par un attachement à droite alors que C_{prec} est attaché dans C_{suiv} par un attachement à gauche.

Lorsque l'analyse est effectuée de droite à gauche, la situation est inversée, le segment de phrase correspondant à C_{suiv} est situé à gauche du segment correspondant à C_{prec}. Ainsi, lors de la réduction, C_{prec} est attaché dans C_{suiv} par un attachement à droite alors que C_{suiv} est attaché dans C_{prec} par un attachement à gauche.

C.2.3 L'opération d'attachement

L'opération d'attachement d'un arbre A dans un arbre B consiste à unifier la racine de A avec un nœud de B appelé site d'attachement. L'opération d'attachement a été décrite en imposant trois conditions à l'arbre résultant. La première impose que celui-ci possède un domaine noir connexe (les nœuds noirs de l'arbre doivent eux-même former un arbre), la seconde impose que l'unification de la racine et du site aboutisse, et la troisième impose que les contraintes linéaires associées à la dépendance d'attachement (la dépendance reliant les domaines noirs des deux arbres, à l'issue d'une opération d'attachement) soient satisfaites. Ces trois conditions vont être vérifiées à tour de rôle. Dans un premier temps, les sites potentiels d'attachement dans B sont déterminés de telle sorte que leur unification avec la racine de A aboutisse à un arbre présentant un domaine noir connexe. Dans un second temps, l'unification de la racine de A avec les différents sites localisés est tentée. Lorsque l'unification aboutit, les contraintes linéaires associées à la dépendance d'attachement sont vérifiées. La vérification consiste à s'assurer que les dépendances sœurs de la dépendance d'attachement sont autorisées, c'est à dire qu'elles aparaîssent dans la liste des frères intérieurs de la dépendance de liason, située au niveau du trait `sep`.

Nous avons repris, dans la figure C.14, les différentes étapes d'une opération d'attachement. Dans un premier temps, les sites d'attachement dans B sont identifiés. Dans un second, temps l'unification de la racine de A et de S est effectuée, elle donne naissance à l'arbre R. Celui-ci est alors substitué au sous-arbre S et les contraintes linéaires sont vérifiées. Un nouvel arbre est alors créé.

A l'instar de l'opération d'unification, l'opération d'attachement est non destructrice : à l'issue de l'attachement, les deux arbres doivent être intacts. C'est pourquoi les opérations décrites ci-dessus sont effectuées sur des copies des arbres à attacher.

C.2.4 La recherche de sites d'attachement

Lors de l'attachement d'un arbre A dans un arbre B, il convient de rechercher dans B les nœuds que l'on essayera d'unifier avec la racine de A. La localisation des sites se fait en fonction de la

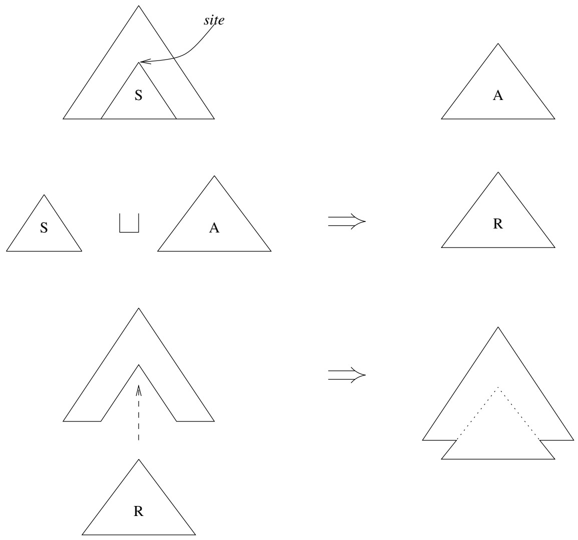


FIG. C.14: Les différentes étapes d'une opération d'attachement

contrainte de connexité du domaine noir de l'arbre qui résultera de l'attachement. Nous avons représenté dans la figure C.15 un cas d'attachement à droite (A correspond à un segment de phrase situé à droite de B). Nous avons identifié dans B les sites potentiels d'attachement : il s'agit de nœuds blancs dépendant directement de nœuds noirs et se trouvant à la droite de ce dernier (la valeur de leur attribut `cote` doit être égale à `d`).

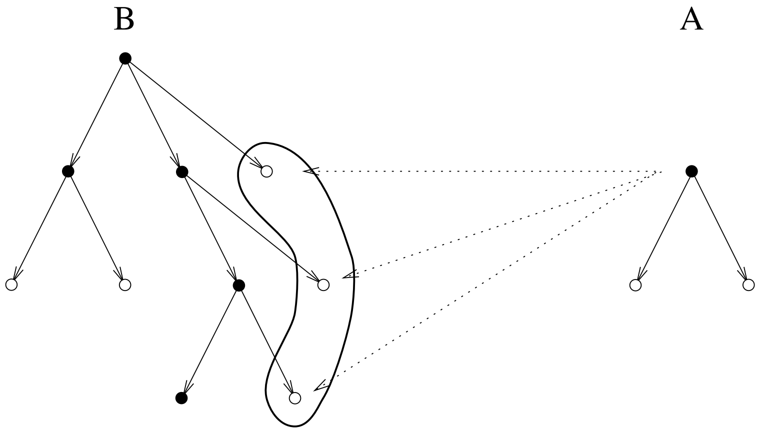


FIG. C.15: Détection de sites d'attachement

Dans l'exemple précédent, la racine de A était constituée d'un nœud noir, ce qui n'est pas toujours le cas, comme le montre la figure C.16 où le premier nœud noir de l'arbre, lorsque ce dernier est parcouru à partir de la racine, se situe à la profondeur deux [4]. Les sites d'attachements sont là aussi recherchés en fonction de la connexité de l'arbre issu de l'attachement. Il s'agit de nœuds noirs possédant au moins un fils noir avec lesquels la racine de A et son fils blanc peuvent s'unifier, assurant la connexité du domaine noir.

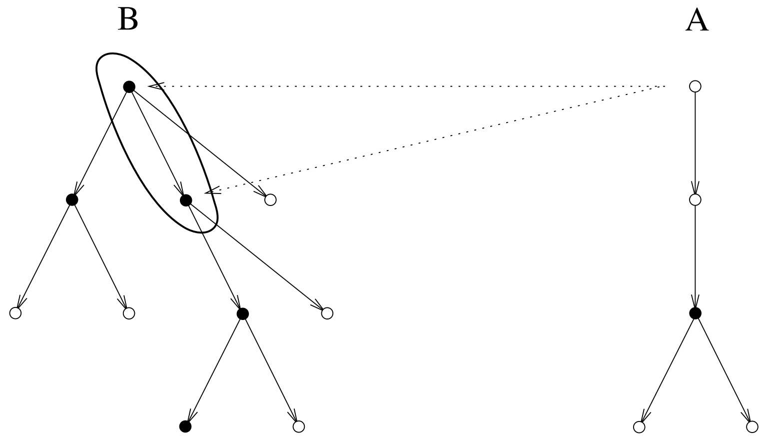


FIG. C.16: Détection de sites d'attachement

La recherche de site d'attachement dépend, de plus, du degré de pseudo-projectivité de la dépendance d'attachement. On pourra remarquer que dans les deux exemples précédents, la recherche des sites d'attachement ne prenait en considération que certains nœuds noirs de B : les nœuds se trouvant sur la branche de l'arbre la plus à droite. Il s'agit d'un cas particulier, où l'arbre issu de l'attachement doit satisfaire à la condition de projectivité. En effet, en unifiant la racine de A avec un nœud situé sur la branche la plus à droite de B ou un de ses dépendants blanc, on a la garantie que la structure générée sera projective. Dans les cas de non projectivité, plus précisément les cas de pseudo-projectivité de degré supérieur à un, il ne suffit plus de considérer la branche la plus à droite de B pour la recherche des sites d'attachement. La figure C.17 représente le domaine noir d'un arbre dans lequel les nœuds ont été regroupés selon le degré de pseudo-projectivité de la structure issue de l'attachement.

Lorsqu'à l'issue d'un rattachement à droite, un nouveau dépendant D est ajouté à un nœud G appartenant à l'ensemble des nœuds du groupe $G2$, le gouverneur linéaire de D[5] sera son ancêtre de niveau deux. Si G appartient au groupe $G3$, le gouverneur linéaire de D sera son ancêtre de niveau trois. Ces deux cas ont été repris dans la figure C.18.

La recherche de sites d'attachement est implémentée par la fonction `sites` qui prend en argument un nœud de B, le type d'attachement dont il s'agit (droite ou gauche), la profondeur de la partie supérieure de A et, finalement, le degré de pseudo-projectivité associé à la dépendance de liaison.

[4]Nous appellerons partie supérieure d'un arbre, la partie située au-dessus du premier nœud noir.

[5]Le gouverneur linéaire d'un nœud X est l'ancêtre de plus haut niveau de X séparant X de son gouverneur.

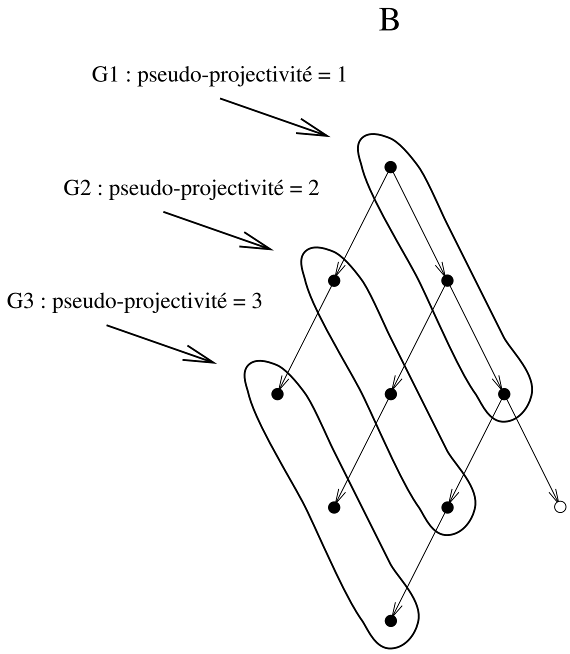


FIG. C.17: Sites potentiels d'attachement regroupés par niveau de pseudo-projectivité

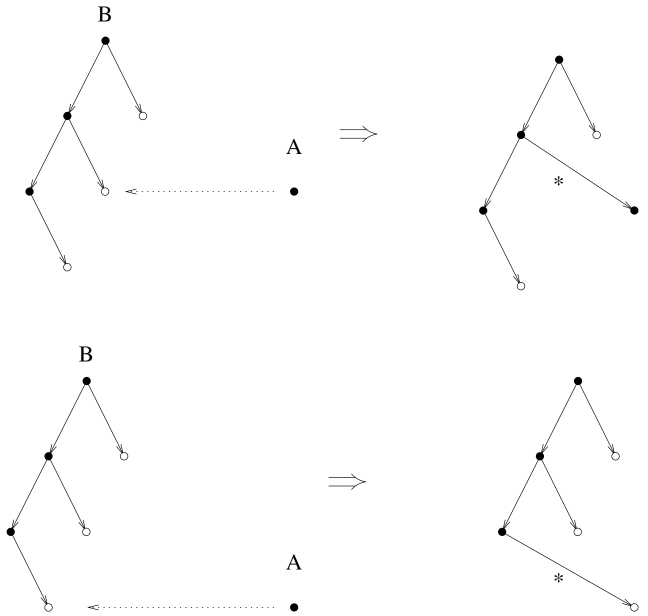


FIG. C.18: Attachements non projectifs

C.2.5 L'algorithme d'analyse

L'algorithme d'analyse se décompose en quatre étapes :

1. Analyse syntaxique gauche-droite

   L'algorithme d'analyse, sans rentrer dans le détail des opérations d'empilement, de réduction et d'attachement, est très simple. Il consiste à lire la phrase de gauche à droite. Après la lecture de chaque mot, les *AES* lui correspondant sont empilés dans la pile graphe et cette dernière est réduite grâce à une opération de réduction droite. L'algorithme s'arrête après lecture du dernier mot et réduction de la pile graphe.

2. Classement des arbres issus de l'analyse dans le tableau des résultats.

   A l'issue de l'analyse gauche-droite, les différents arbres contenus dans la pile graphe sont rangés dans le tableau d'analyse. Un arbre représentant le segment de phrase s'étendant du mot i au mot j sera rangé dans la case i, j. Dans les cas d'ambiguïté, où plusieurs arbres correspondent à un même segment, la case comportera autant d'arbres que d'analyses. Seront représentées dans le tableau les analyses globales aussi bien que les analyses locales. Il faut noter qu'un arbre correspondant à une analyse locale ne s'identifie pas à un sous-arbre d'une analyse totale représentée dans le tableau, mais à une analyse différente, n'ayant pas abouti à une analyse globale. Les arbres sont stockés dans le tableau sous forme linéaire, afin de limiter l'encombrement mémoire et de faciliter la comparaison entre arbres qui aura lieu à l'issue de l'analyse droite-gauche.

3. Analyse droite-gauche

   L'algorithme d'analyse gauche-droite est très proche de celui de l'analyse droite-gauche. Il consiste à lire la phrase de droite à gauche. Après la lecture d'un mot, les arbres élémentaires lui correspondant sont empilés dans la pile graphe et celle-ci est réduite grâce à une opération de réduction gauche alors que dans le cas de l'analyse gauche-droite, il s'agissait d'une réduction droite. L'algorithme s'arrête après la lecture et l'empilement du premier mot de la phrase et réduction de la pile graphe.

4. Classement des arbres issus de l'analyse dans le tableau des résultats.

   Les arbres construits durant la seconde analyse sont stockés dans le même tableau d'analyse, ce qui permet, lors du stockage, de comparer les arbres. Nous avons vu, en effet, que certaines analyses seront construites deux fois : la première lors de l'analyse gauche-droite et la seconde lors de l'analyse droite-gauche. C'est pourquoi, lors du stockage d'un arbre correspondant au segment de phrase i, j, issu de la seconde analyse, il sera comparé aux arbres de la case i, j déjà stockés dans le tableau, de façon à ne pas représenter deux fois un même arbre dans le tableau. La comparaison des arbres représentés sous forme linéaire est triviale, elle se résume à une comparaison de listes, du fait que ces dernières sont triées.

Table des figures

Bibliographie

[Abeillé, 1991] Abeillé, A. (1991). *Une grammaire lexicalisée d'arbres adjoints pour le français. Applications à l'analyse automatique.* Thèse de doctorat, Université Paris 7.

[Abeillé, 1993] Abeillé, A. (1993). *Les nouvelles syntaxes. Grammaires d'unification et analyse du français.* Linguistique. Armand Colin.

[Abeillé et al., 1990] Abeillé, A., Schabes, Y., & Joshi, A. (1990). *Using lexicalized* TAG*s for Machine Translation.* Edité dans *Proceedings of the 13th International Conference on Computational Linguistics (COLING'90)*, pages 1–6, Helsinki.

[Adriaens, 1994] Adriaens, G. (1994). *Simplified English Grammar and Style Correction in an* MT *Framework : the* LRE SECC *Project.* Edité dans *Proceedings of the 16th Conference on Translating and the Computer*, pages 78–88, Londres.

[Adriaens & Schreurs, 1992] Adriaens, G. & Schreurs, D. (1992). *From* COGRAM *to* ALCOGRAM *: toward a Controlled English Grammar Checker.* Edité dans *Proceedings of the 14th International Conference on Computational Linguistics (COLING'92)*, pages 595–601, Nantes, France.

[AECMA, 1989] AECMA (1989). *A guide for the preparation of aircraft maintenance documentation in the international aerospace maintenance language.* Association Européenne des Constructeurs de Matériel Aérospatial, Paris.

[Aït-Kaci & Nasr, 1986] Aït-Kaci, H. & Nasr, R. (1986). LOGIN*: a logic programming language with built-in inheritance. The Journal of Logic Programming*, 3:185–215.

[Almqvist & Hein, 1996] Almqvist, I. & Hein, A. S. (1996). *Definig* SCANIA *Swedish - a controlled language for truck maintenance.* Edité dans *International Workshop on Controlled Language Applications*, pages 159–165, Louvain, Belgique.

[Alonso Ramos & Tutin, 1993] Alonso Ramos, M. & Tutin, A. (1993). *Les fonctions lexicales du dictionnaire explicatif combinatoire pour l'étude de la cohésion lexicale. Linguisticae Investigationes*, XVII(1):161–188.

[ALPAC, 1966] ALPAC (1966). *Language and machines: computer in translation and linguistics.* Automatic Language Processing Advisory Committee, National Academy of Sciences, Washington.

[Bennett & Slocum, 1985] Bennett, W. & Slocum, J. (1985). *The* LRC *machine translation system. Computational Linguistics*, 11:111–121.

[Béringer, 1988] Béringer, H. (1988). *Traitement Automatique des Ambiguïtés Structurales du Français utilisant la Théorie Sens-Texte.* Thèse de doctorat, Université Paris 6.

[Boyer & Lapalme, 1985] Boyer, M. & Lapalme, G. (1985). *Generating paraphrases from meaning-text semantic networks. Computational Intelligence.*

[Bresnan, 1982] Bresnan, J., éditeur (1982). *The Mental Representation of Grammatical Relations.* MIT Press.

[Bröker et al., 1994a] Bröker, N., Hahn, U., & Schacht, S. (1994a). *Concurrent Lexicalized Dependency Parsing: a Behavioral view on Parse Talk Events.* Edité dans *Proceedings of the 16th International Conference on Computational Linguistics (COLING'94)*, pages 489–493, Tokyo.

[Bröker et al., 1994b] Bröker, N., Hahn, U., & Schacht, S. (1994b). *Concurrent Lexicalized Dependency Parsing: the Parse Talk Model.* Edité dans *Proceedings of the 16th International Conference on Computational Linguistics (COLING'94)*, pages 379–385, Tokyo.

[Candito, 1995] Candito, M.-H. (1995). *La hiérarchisation des structures élémentaires d' une grammaire lexicalisée d'arbres adjoints pour le français.* Mémoire de DEA de linguistique. Université Paris 7.

[Chevalier, 1978] Chevalier, M. (1978). TAUM-METEO *: description du système.* Groupe de recherche en traduction automatique, Université de Montréal.

[Chomsky, 1981] Chomsky, N. (1981). *Lectures in Government and Binding.* Studies in generative grammar 9. Foris, Dordrecht.

[Clémencin, 1996] Clémencin, G. (1996). *Integration of a* CL*-checker in an Operational* SGML *Authoring Environment : Methodological and Technical Issues.* Edité dans *International Workshop on Controlled Language Applications*, pages 32–41, Louvain, Belgique.

[Cunningham & Cohen, 1984] Cunningham, D. & Cohen, G. (1984). *Creating Technical Manuals:a step by step approach to writing user-friendly instructions.* Mc Graw Hill Book Company, New York.

[Danlos, 1985] Danlos, L. (1985). *Génération automatique de textes en langues naturelles.* Etudes et recherches en informatique. Masson.

[Danlos, 1992] Danlos, L. (1992). *Support verb constructions: linguistic properties, representation, translation. Journal of French Language Studies*, 2(1):1–32.

[de Smedt, 1990] de Smedt, K. (1990). Incremental sentence generation. Rapport Technique NICI 90-01, Nijmegen Institute for Cognition Research and Information Technology.

[DiMarco & Hirst, 1988] DiMarco, C. & Hirst, G. (1988). *Stylistic Grammars in Language Translation.* Edité dans *Proceedings of the 12th International Conference on Computational Linguistics (COLING'88)*, volume I, pages 148–153, Budapest.

[EUROTRA, 1991] EUROTRA (1991). *The Eurotra linguistic specifications.* Commission of the European Communities, studies in machine translation and natural language processing ' edition. Catalogue no CD-AM-91-001-EN-C.

[Fillmore, 1968] Fillmore, C. (1968). *The case for case.* Universals in Linguistic Theory. Holt and Rinehart and Winston, New-York.

[Flickinger, 1987] Flickinger, D. (1987). *Lexical Rules in the Hierarchical Lexicon.* Thèse de doctorat, Univeristé de Stanford.

[Fraser, 1989] Fraser, N. (1989). *Parsing and dependency grammar. University College London Working Papers in Linguistics*, 1:296–319.

[Fraser, 1993] Fraser, N. (1993). *Dependency Parsing.* Thèse de doctorat, University College London.

[Fraser & Hudson, 1992] Fraser, N. & Hudson, R. (1992). *Inheritance in Word Grammar. University College London Working Papers in Linguistics*, 18(2):133–138.

[Fuchs, 1982] Fuchs, C. (1982). *La paraphrase.* Linguistique nouvelle. Presses Universitaires de France.

[Gaifman, 1965] Gaifman, H. (1965). *Dependency Systems and Phrase-Structure Systems. Information and Control*, 8:304–337.

[Gazdar et al., 1985] Gazdar, G., Klein, E., Pullum, G. K., & Sag, I. (1985). *Generalized Phrase Structure Grammar.* Harvard University Press.

[Gazdar & Mellish, 1989] Gazdar, G. & Mellish, C. (1989). *Natural Language Processing in LISP.* Addison-Wesley.

[Genthial, 1991] Genthial, D. (1991). *Contribution à la construction d'un système robuste d'analyse du français.* Thèse de doctorat, Université Joseph Fourier, Grenoble, France.

[Genthial et al., 1990] Genthial, D., Courtin, J., & Kowarski, I. (1990). *Contribution of a category hierarchy to the robustness of syntactic parsing.* Edité dans *Proceedings of the 13th International Conference on Computational Linguistics (COLING'90)*, pages 139–144, Helsinki.

[GIFAS, 1990] GIFAS (1990). *Guide du rédacteur.* Groupement des Industries Françaises Aéronautiques et Spatiales, Paris, France.

[Gladkij & Mel'čuk, 1975] Gladkij, A. & Mel'čuk, I. A. (1975). *Tree grammars* I. *A formalism for syntactic transformations in natural languages. Linguistics*, 150:47–82.

[Goyvaerts, 1996] Goyvaerts, P. (1996). *Controlled English, curse or blessing? A user's perspective.* Edité dans *International Workshop on Controlled Language Applications*, pages 137–142, Louvain, Belgique.

[Grinberg et al., 1995] Grinberg, D., Lafferty, J., & Sleator, D. D. (1995). A robust parsing algorithm for link grammars. Rapport Technique CMU-CS-95-125, School of Computer Science, Carnegie Mellon University.

[Grishman & Kittredge, 1986] Grishman, R. & Kittredge, R., éditeurs (1986). *Analysing Language in Restricted Domains.* Lawrence Erlbaum Associates, Hillsdale, New Jersey, Etats-Unis.

[Harris, 1968] Harris, Z. (1968). *Mathematical Structures of Language.* Wiley, New-York.

[Hayashi, 1992] Hayashi, Y. (1992). *A three-level Revision Model for Improving Japanese Bad Styled Expressions.* Edité dans *Proceedings of the 14th International Conference on Computational Linguistics (COLING'92)*, Nantes, France.

[Hayes et al., 1996] Hayes, P., Maxwell, S., & Schmandt, L. (1996). *Controlled English Advantages for Translated and Original English Document.* Edité dans *International Workshop on Controlled Language Applications*, pages 84–92, Louvain, Belgique.

[Hays, 1964] Hays, D. G. (1964). *Dependency Theory: A Formalism and some Observations. Language*, 40:511–525.

[Hellwig, 1985] Hellwig, P. (1985). PLAIN: Program for language analysis and inference. Rapport technique, Computing Unit and Linguistics and International Studies Department, University of Surrey, Guilford, Angleterre.

[Hellwig, 1986a] Hellwig, P. (1986a). *Dependency Unification Grammar.* Edité dans *Proceedings of the 11th International Conference on Computational Linguistics (COLING'86)*, Bonn.

[Hellwig, 1986b] Hellwig, P. (1986b). *The* PLAIN *Approach to Natural Language Processing.* Edité dans *Spring meeting of the SHARE European Association (SEAS).*

[Hellwig, 1988] Hellwig, P. (1988). *Chart Parsing According to the slot and filler principle.* Edité dans *Proceedings of the 12th International Conference on Computational Linguistics (COLING'88)*, pages 242–244, Budapest.

[Hudson, 1984] Hudson, R. (1984). *Word Grammar.* Basil Blackwell, Oxford, Angleterre.

[Hudson, 1989] Hudson, R. (1989). *Towards a computer-testable word grammar of English. University College London Working Papers in Linguistics*, 1:321–338.

[Hutchins & Somers, 1992] Hutchins, J. W. & Somers, H. L. (1992). *An Introduction to Machine Translation.* Academic Press, Cambridge.

[Isabelle, 1978] Isabelle, P. (1978). TAUM-AVIATION *: un système automatique pour la traduction automatique de manuels techniques.* Edité dans *Proceedings of the 7th International Conference on Computational Linguistics (COLING'78).*

[Jäppinen et al., 1986] Jäppinen, H., Lethola, A., & Valkonen, K. (1986). *Functional Structures for Parsing Dependency Constraints.* Edité dans *Proceedings of the 11th International Conference on Computational Linguistics (COLING'86)*, pages 461–463, Bonn.

[Jäppinen et al., 1987] Jäppinen, H., Lethola, A., & Valkonen, K. (1987). *Blackboard-Based Dependency Parsing.* Edité dans *10th Intenational Joint Conference on Artificial Intelligence*, pages 700–702, Milan, Italie.

[Karlsson, 1995] Karlsson (1995). *Constraint Grammar : a language Independent system for Parsing Unrestricted text.* Mouton de Gruyter.

[Kasper, 1987] Kasper, R. T. (1987). *A Unification Method for Disjunctive Feature Description.* Edité dans *25th Meeting of the Association for Computational Linguistics (ACL'87)*, pages 235–242.

[Kay, 1984] Kay, M. (1984). *Functional unification grammar: a formalism for machine translation.* Edité dans *Proceedings of the 10th International Conference on Computational Linguistics (COLING'84)*, pages 75–78, Stanford, Etats-Unis.

[Kittredge, 1982] Kittredge, R. (1982). *Variation and Homogeneity of Sublanguages.* Edité dans *Analysing Language in Restricted Domains*, chapitre 4, pages 81–106. Walter de Gruyter, Berlin.

[Kocourek, 1991] Kocourek, R. (1991). *La langue française de la technique et de la science.* Brandstetter Verlag.

[Lafferty et al., 1992] Lafferty, J., Sleator, D. D., & Temperley, D. (1992). Grammatical trigrams: A probabilistic approach of link grammars. Rapport Technique CMU-CS-92-181, School of Computer Science, Carnegie Mellon University.

[Lazard, 1994] Lazard, G. (1994). *L'actance.* Linguistique nouvelle. Presses Universitaires de France.

[Lecerf, 1960] Lecerf, Y. (1960). *Programme des conflits, modèle des conflits. Bulletin bimestriel de l'*ATALA, 4,5.

[Lehrberger, 1982] Lehrberger, J. (1982). *Automatic Translation and the Concept of Sublanguage.* Edité dans Kittredge, R. & Lehrberger, J., éditeurs, *Analysing Language in Restricted Domains*, chapitre 3, pages 81–106. Walter de Gruyter, Berlin.

[Lerat, 1995] Lerat, P. (1995). *Les langues spécialisées.* Linguistique nouvelle. Presses Universitaires de France.

[Lesmo & Lombardo, 1992] Lesmo, L. & Lombardo, V. (1992). *The Assignment of Grammatical Relations in Natural Language Processing.* Edité dans *Proceedings of the 14th International Conference on Computational Linguistics (COLING'92)*, pages 1090–1094, Nantes, France.

[Lombardo, 1995] Lombardo, V. (1995). *Parsing and Recovery.* Edité dans *17th Annual Conference of the Cognitive Science Society*, pages 648–653, Pittsburgh.

[Lux & Dauphin, 1996] Lux, V. & Dauphin, E. (1996). *Corpora Studies : a Contribution to the Definition of a Controlled Language.* Edité dans *International Workshop on Controlled Language Applications*, pages 193–204, Louvain, Belgique.

[Marcus, 1965] Marcus, S. (1965). *Sur la notion de projectivité. Zeitschr. f. math. Logik und Grundlagen d. Math.*, 11:181–192.

[Means & Godden, 1996] Means, L. & Godden, K. (1996). *The Controlled Automotive Service Language (*CASL*) Project.* Edité dans *International Workshop on Controlled Language Applications*, pages 106–114, Louvain, Belgique.

[Mel'čuk et al., 1987] Mel'čuk, I., Arbatchewsky-Jumarie, N., Dagenais, L., Lenitsky, L., Iordanskaja, L., Lefebvre, M.-N., & Mantha, S. (1987). *Dictionnaire explicatif et combinatoire du français contemporain - Recherches lexico-sémantiques II.* Les Presses de l'Université de Montréal.

[Mel'čuk et al., 1992] Mel'čuk, I., Arbatchewsky-Jumarie, N., Iordanskaja, L., & Mantha, S. (1992). *Dictionnaire explicatif et combinatoire du français contemporain - Recherches lexico-sémantiques III.* Les Presses de l'Université de Montréal.

[Mel'čuk et al., 1984] Mel'čuk, I., Arbatchewsky-Jumarie, N., Lenitsky, L., Iordanskaja, L., & Lessard, A. (1984). *Dictionnaire explicatif et combinatoire du français contemporain - Recherches lexico-sémantiques I.* Les Presses de l'Université de Montréal.

[Mel'čuk, 1988a] Mel'čuk, I. A. (1988a). *Dependency Syntax: Theory and Practice.* State University of New York Press, New York.

[Mel'čuk, 1988b] Mel'čuk, I. A. (1988b). *Paraphrase et lexique dans la théorie linguistique Sens-Texte. Cahiers de lexicologie*, LII.

[Mel'čuk & Pertsov, 1987] Mel'čuk, I. A. & Pertsov, N. V. (1987). *Surface Syntax of English.* John Benjamins, Amsterdam/Philadelphia.

[Mel'čuk & Polguère, 1987] Mel'čuk, I. A. & Polguère, A. (1987). *A Formal Lexicon in the Meaning-Text Theory. Computational Linguistics*, 13(3-4):13–54.

[Mel'čuk & Zholkovsky, 1970] Mel'čuk, I. A. & Zholkovsky, A. (1970). *Towards a functioning meaning-text model of language. Linguistics*, 57:10–47.

[Polguère, 1990] Polguère, A. (1990). *Structuration et mise en jeu procédurale d'un modèle linguistique déclaratif dans un cadre de génération de texte.* Thèse de doctorat, Université de Montréal.

[Polguère, 1992] Polguère, A. (1992). *Remarks on Meaning-Text semantic networks.* Edité dans *International Workshop on the Meaning-Text Theory*, pages 99–108, Darmstadt, Allemagne.

[Pollard & Sag, 1994] Pollard, C. & Sag, I. (1994). *Head-driven Phrase Structure Grammmar.* CSLI Series. University of Chicago Press.

[Pustejovsky, 1991] Pustejovsky, J. (1991). *The Generative Lexicon. Computational Linguistics*, 17(4):409–441.

[Pustejovsky, 1995] Pustejovsky, J. (1995). *The Generative Lexicon.* MIT Press, Cambridge.

[Robinson, 1970] Robinson, J. J. (1970). *Dependency structures and transformational rules. Language*, 46(2):259–285.

[Sabah, 1988] Sabah, G. (1988). *L'intelligence artificielle et le langage.* Hermès.

[Sager, 1982] Sager, N. (1982). *Syntactic Formatting of Science Information.* Edité dans Kittredge, R. & Lehrberger, J., éditeurs, *Analysing Language in Restricted Domains*, pages 9–26. Walter de Gruyter.

[Sager, 1986] Sager, N. (1986). *Sublanguage: Linguistic Phenomenon, Computational Tool.* Edité dans Grishman, R. & Kittredge, R., éditeurs, *Analysing Language in Restricted Domains: sublanguage description and processing*, pages 1–18. Lawrence Erlbaum Associates.

[Schabes, 1990] Schabes, Y. (1990). *Computational and Mathematical Properties of Lexicalized Grammars.* Thèse de doctorat, Université de Pennsylvanie, Philadelphie, Etats-Unis.

[Schabes et al., 1988] Schabes, Y., Abeillé, A., & Joshi, A. (1988). *Parsing Strategies with Lexicalized Grammars : Tree Adjoining Grammars.* Edité dans *Proceedings of the 12th International Conference on Computational Linguistics (COLING'88)*, volume 2, pages 578–583, Budapest.

[Schachtl, 1996] Schachtl, S. (1996). *Requirement for Controlled German in Industrial Applications.* Edité dans *International Workshop on Controlled Language Applications*, pages 143–149, Louvain, Belgique.

[Sgall, 1992] Sgall, P. (1992). *Valency and Underlying Structure.* Edité dans *International Workshop on The Meaning Text Theory*, pages 15–35, Darmstadt, Allemagne.

[Shieber, 1986] Shieber, S. (1986). *An introduction to unification based approaches to grammars.* CSLI Series. Chicago Press.

[Shieber & Schabes, 1990] Shieber, S. & Schabes, Y. (1990). *Synchronous Tree-Adjoining Grammars.* Edité dans *Proceedings of the 13th International Conference on Computational Linguistics (COLING'90)*, pages 253–258, Helsinki.

[Sleator & Temperley, 1991] Sleator, D. D. & Temperley, D. (1991). Parsing english with a link grammar. Rapport Technique CMU-CS-91-196, School of Computer Science, Carnegie Mellon University.

[Slocum, 1987] Slocum, J. (1987). METAL*: The* LRC *Machine Translation System.* Edité dans King, M., éditeur, *Machine Translation Today: the State of the Art*, pages 319–350. Edinburgh University Press.

[Starosta, 1975] Starosta, S. (1975). *Case in the Lexicon.* Edité dans *11th International Congress of Linguists*, pages 805–813.

[Starosta & Nomura, 1986] Starosta, S. & Nomura, H. (1986). *Lexicase Parsing : A Lexicon-driven Approach to Syntactic Analysis.* Edité dans *Proceedings of the 11th International Conference on Computational Linguistics (COLING'86)*, Bonn.

[Tesnière, 1959] Tesnière, L. (1959). *Eléments de syntaxe structurale.* Klincksieck, Paris.

[Thurmair, 1990] Thurmair, G. (1990). *Parsing for Grammar and Style Checking.* Edité dans *Proceedings of the 13th International Conference on Computational Linguistics (COLING'90)*, pages 365–370, Helsinki.

[Tomita, 1985] Tomita, M. (1985). *Efficient Parsing for Natural Language.* Kluwer Academic Publishers, Boston, MA.

[Tomita, 1987] Tomita, M. (1987). *An Efficient Augmented Context Free Parsing Algorithm. Computational Linguistics*, 13(1-2):31–46.

[Tomita, 1988] Tomita, M. (1988). *Graph Structured Stack and Natural Language Parsing.* Edité dans *26th Meeting of the Association for Computational Linguistics (ACL'88)*, Buffalo, NY.

[van der Eijck et al., 1996] van der Eijck, P., de Koning, M., & van der Steen, G. (1996). *Controlled Language Correction and Translation.* Edité dans *International Workshop on Controlled Language Applications*, pages 64–73, Louvain, Belgique.

[Wieringa et al., 1992] Wieringa, D., Moore, C., & Barnes, V. (1992). *Procedure Writing - principles and practices-.* Batelle Press, Ohio.

[Wojcik et al., 1990] Wojcik, R. H., Hoard, J. E., & Holzhauser, K. (1990). *The* BOEING *Simplified English Checker.* Edité dans *Proceedings of the International Conference, Human Machine Interaction and Artificial Intelligence in Aeronautics and Space*, pages 38–45, Toulouse : Centre d'Etude et de Recherches de Toulouse, France.

[Wojcik & Holmback, 1996] Wojcik, R. H. & Holmback, H. (1996). *Getting a Controlled Language off the Ground at* BOEING*.* Edité dans *International Workshop on Controlled Language Applications*, pages 22–31, Louvain, Belgique.

[Zajac & Heald, 1996] Zajac, R. & Heald, I. (1996). *Syntactic and Semantic Problems in the Use of a Controlled Language.* Edité dans *International Workshop on Controlled Language Applications*, pages 205–215, Louvain, Belgique.

Index