De l'étiquetage syntaxique pour les grammaires catégorielles de dépendances à l'analyse par transition dans le domaine de l'analyse en dépendances non-projective

Ophélie Lacroix

l'université nantes angers le mans
COMMUNAUTÉ D'UNIVERSITÉS ET ÉTABLISSEMENTS

UNIVERSITÉ DE NANTES

Thèse de Doctorat

Ophélie LACROIX

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université de Nantes
sous le label de l'Université de Nantes Angers Le Mans

École doctorale : Sciences et technologies de l'information, et mathématiques

Discipline : Informatique et Linguistique, section CNU 27
Unité de recherche : Laboratoire d'informatique de Nantes-Atlantique (LINA)

Soutenue le 8 décembre 2014

De l'étiquetage syntaxique pour les grammaires catégorielles de dépendances à l'analyse par transition dans le domaine de l'analyse en dépendances non-projective

JURY

Président :	**M. Christian RETORÉ**, Professeur des Universités, Université Montpellier 2
Rapporteurs :	**M. Matthieu CONSTANT**, Maître de Conférences HDR, Université Paris-Est Marne-la-Vallée
	M. Alexis NASR, Professeur des Universités, Université Aix Marseille
Examinateur :	**M^me^ Marie CANDITO**, Maître de Conférences, Université Paris Diderot
Directeur de thèse :	**M. Colin DE LA HIGUERA**, Professeur des Universités, Université de Nantes
Co-encadrant de thèse :	**M. Denis BÉCHET**, Maître de Conférences, Université de Nantes

Remerciements

Faire une thèse consiste tout d'abord à comprendre ce qu'est un travail de recherche, puis à chercher... et à trouver, et surtout à apprendre de nouvelles choses encore et toujours. Il s'agit bien sûr d'un métier mais également de l'aboutissement de huit années d'études. C'est durant mes 3 années (et quelques mois) de thèse que j'ai eu l'occasion de découvrir le monde de la recherche et de rencontrer et d'apprendre à connaître un grand nombre de personnes qui m'ont apportée beaucoup personnellement et professionnellement.

En tout premier lieu, je souhaite remercier mon premier directeur de thèse Alexandre Dikovsky, Sacha, qui m'a encadrée malgré l'état difficile dans lequel il fut. Nous n'avons pas eu l'occasion de nous voir très souvent après ma première année de thèse mais nos échanges ont toujours été très fructueux. Il a toujours été là pour répondre à mes questions, pour m'orienter dans ma recherche et pour apporter des critiques pertinentes sur mes travaux. J'admire les travaux de recherche qu'il a mené durant sa vie et je le remercie de m'avoir fait confiance pour entamer et mener à bien ces travaux de thèse. J'espère qu'il aurait été fier de l'aboutissement vers lequel j'ai conduit ces travaux.

Je souhaite, en second, remercier mes encadrants actuels, Denis Béchet et Colin de la Higuera. Je remercie Denis avec qui j'ai beaucoup travaillé durant mes trois années de thèse. Nous avons explorés différents champs de recherche. Les travaux que nous avons effectués m'ont beaucoup apportée d'un point de vue scientifique. Nous avons également dispensé des heures de travaux pratiques ensemble à l'Université de Nantes et j'ai ainsi eu la chance de pouvoir donner des cours qui m'intéressait particulièrement dans le domaine qui m'avait initialement motivée à poursuivre des études orientées recherche. Je remercie Colin pour avoir accepté de reprendre la direction de ma thèse à la moitié de ma troisième année. Ce n'était pas évident de se plonger dans des travaux de thèse (presque) finalisés dans un domaine assez spécialisé. Nous n'avons pas eu le temps de collaborer sur mes travaux de thèse mais Colin m'a rapidement donné de bons conseils quant à la rédaction de mon mémoire et à la poursuite de ma carrière dans la recherche.

Je souhaite également adresser mes remerciements aux rapporteurs et aux examinateurs de ma thèse pour avoir accepté d'être les membres de mon jury et avoir été présents lors de ma soutenance. Je remercie mes rapporteurs, Matthieu Constant et Alexis Nasr, pour avoir pris le temps de lire consciencieusement ce mémoire et d'écrire chacun un rapport détaillé de mes travaux avec des remarques très intéressantes qui m'ont permis de me poser encore de nouvelles questions pour aller toujours plus loin dans la recherche de solutions judicieuses. Je remercie de la même manière mes examinateurs, Marie Candito et Christien Retoré, pour m'avoir apportée également leurs avis sur mes travaux à travers leurs questions et remarques pertinentes lors de ma soutenance.

Par ailleurs, je remercie l'équipe TALN et tous ces membres (doctorants et permanents) pour m'avoir accueillie au sein du LINA et avec qui j'ai passé une très bonne semaine aux Sables-d'Olonne pour l'organisation de la conférence TALN 2013. Je remercie notre responsable d'équipe préférée, Béatrice Daille, qui prend soin de chacun d'entre nous. Et je souhaite en particulier remercier Florian Boudin pour tous les bons conseils qu'il m'a donnés lorsque nous avons entrepris de travailler ensemble, pour être toujours disponible quand on a besoin d'une relecture et pour avoir égayé mes repas du midi à la cafét.

Je remercie le personnel administratif et le service informatique pour tout le travail qu'ils font pour nous rendre la vie plus facile au labo, pour être toujours disponibles quand on a des soucis (techniques ou personnels), quand on part en mission, et tout simplement parce qu'ils sont des éléments essentiels au bon fonctionnement et à la vie du laboratoire. Je pense ici à Annie, Floriane, Sabine... et je souhaite remercier en particulier Anne-Françoise Quin, qui m'a écoutée lorsque j'ai eu des soucis et qui s'est montrée particulièrement soucieuse vis-a-vis de ma situation. Je remercie par la même occasion Charlotte Truchet qui a également su être attentive dès le début et me donner de bons conseils pour surmonter mes problèmes, et qui défend et prend en main des sujets importants au sein de l'université.

Bien sûr, je remercie également l'association Login et ses membres, ceux avec qui j'ai partagé beaucoup de temps durant ma thèse, mes collègues de bureau Liza et Adrien, nos presque voisins de bureau et fournisseurs de chocolat préférés Benjamin et Audrey. Il y a également les anciens doctorants, Thomas, Laurent, Amir, Mohamed, Rima, Olivier, Marie, Prajol, les encore plus anciens, Aurélien, Matthieu, Fabien. Il y a également les nouveaux doctorants, je ne citerai pas tout le monde mais je souhaite du courage à Firas et aux nouveaux doctorants de l'équipe, anciens étudiants de master, Hugo, Soufian, Grégoire et Joseph.

Je remercie mes amis, les très vieux amis que j'essaie de revoir dès que je peux et ceux que j'ai rencontrés durant mes études supérieures. Je remercie en particulier Noémie, avec qui je flâne régulièrement en conférence et qui sans faire exprès m'a laissée la place pour pouvoir candidater à mon sujet de thèse actuel.

Je remercie ma famille ; mes parents, mes soeurs, mes grand-parents ; qui essaient d'élucider le mystère de mon sujet de thèse à chaque fois qu'on parle de mon « travail », qui surtout m'ont toujours soutenus dans les choix que j'ai fait pour arriver jusque là, qui, je le sais, sont fiers de moi et qui, même si j'ai trop travaillé, ont compris que parfois je ne pouvais pas être entièrement disponible.

La thèse, c'est également un travail sur soi-même dans lequel on peut passer très rapidement de l'enthousiasme ou découragement. Alors je remercie pour finir l'homme qui partage ma vie au quotidien depuis déjà longtemps, Nicolas, pour être toujours à mes côtés quelque soit mon humeur, pour m'avoir suivie quand j'ai choisi de poursuivre mes études, pour avoir toujours été aux petits soins quel que fut mon état de stress, de découragement, de fatigue... mais surtout pour accompagner mes moments de bonheur.

Table des matières

Liste des tableaux

Table des figures

À Sacha,

1

Introduction

1.1 Contexte

Combinant informatique et linguistique, le traitement automatique des langues est un domaine vaste qui englobe des problèmes très différents mais souvent connectés. L'analyse syntaxique fait partie des tâches qui ont été et sont toujours largement étudiées pour elle-mêmes de par leur intérêt linguistique mais qui sont également utilisées pour traiter d'autres problèmes du domaine tels que l'analyse sémantique, la traduction, la modélisation du langage, la recherche d'informations, la correction automatique, etc.

L'analyse syntaxique, en tant qu'étape parmi ces différentes tâches du traitement automatique des langues, a pour objectif de construire des représentations syntaxiques à partir de phrases du langage naturel, c'est à dire de bâtir des relations syntaxiques correctes entre les mots ou les segments de texte. Cette tâche peut s'appliquer à des phrases du langage écrit (e.g. textes littéraires) ou parlé (e.g. transcriptions de l'oral), sur des phrases grammaticalement correctes ou non (par exemple pour être utilisé comme outil de vérification).

À travers les nombreux travaux traitant de l'analyse syntaxique, on voit que le problème peut être étudié selon deux axes : l'analyse syntagmatique ou l'analyse en dépendances. D'une part, l'analyse syntaxique par syntagmes (ou constituants) consiste à annoter et regrouper les mots d'une phrase par groupes syntaxiques, que l'on appelle syntagmes[1]. En ce cas, chaque mot dispose d'une fonction grammaticale. Puis, les mots sont regroupés en syntagmes. Un syntagme étant un groupe auquel est assigné un rôle syntaxique de telle manière que ce groupe peut être remplacé par un autre groupe ayant le même rôle syntaxique sans que la grammaticalité de la phrase en soit altérée. D'un autre côté, l'analyse syntaxique en dépendances vise à établir des relations binaires entre les mots d'une phrase. Chaque mot de la phrase est gouverné par un unique autre mot (excepté la racine de la phrase) : ils sont liés par une relation gouverneur-dépendant ayant un rôle syntaxique spécifique. L'ensemble de ces relations (i.e. dépendances) pour une phrase donnée forme une structure de dépendances.

Cette représentation syntaxique a tout d'abord été mise en avant par Tesnière (1959) pour le français. Puis la théorie Sens-Texte de Mel'cuk (1988) a permis d'éclaircir l'utilité syntaxique et

[1] Un syntagme est une unité syntaxique intermédiaire entre le mot et la phrase, un groupe d'un ou plusieurs mots comprenant un noyau et de potentiels compléments e.g. groupe nominal, groupe verbal, etc.

sémantique de la représentation en dépendances. Il met particulièrement en opposition représentation en dépendances et représentation par syntagmes. L'avantage des dépendances est évident pour les langues comme le tchèque ou le russe qui acceptent une certaine flexibilité dans l'ordre des mots. Cependant, pour Mel'cuk, les structures de dépendances sont aussi plus informatives que les structures par constituants. Les structures par constituants indiquent seulement le rôle local de chaque mot ou groupe de mots dans une phrase tandis que les structures de dépendances permettent de lier chaque mot dans la phrase au(x) mot(s) avec le(s)quel(s) il a une relation syntaxique particulière et donc potentiellement de préciser des relations entre des mots distants dans la phrase. Ces relations de longue distance concernent donc aussi bien les langues comme le français ou l'anglais. Par conséquent, lorsque l'on souhaite conserver l'ordre des mots dans une phrase lors d'une analyse syntaxique, on doit faire face aux problèmes de discontinuité. Une discontinuité apparaît dans une phrase lorsqu'un segment de texte est détaché de son contexte syntaxique direct, c'est à dire lorsque des éléments syntaxiquement liés sont distants dans la phrase et séparés par des mots qui ne leurs sont pas liés.

Les taux de discontinuités dans les langues naturelles sont faibles si l'on considère l'ensemble des relations syntaxiques. Cependant ces discontinuités apparaissent régulièrement dans les phrases et concernent des cas redondants. Des méthodes permettent de passer outre ces difficultés en traitant localement les liens syntaxiques et en ignorant les potentielles relations sémantiques qui y sont liées. Elles permettent d'obtenir des représentations en dépendances projectives (dont les dépendances ne se croisent pas) et sont plus simples à traiter (complexité algorithmique réduite). Une autre solution est de gérer les discontinuités des langues en acceptant les dépendances non-projectives[2] dans les représentations en dépendances. Dans cette thèse nous nous intéressons essentiellement aux représentations en dépendances autorisant les dépendances non-projectives.

1.2 Présentation de la problématique

Dans le domaine de l'analyse syntaxique deux approches différentes sont souvent mises en opposition : les systèmes à base de règles et les systèmes dirigés par les données. Bien que les deux approches soient compatibles, de plus en plus de travaux étudiant des méthodes pour l'analyse en dépendances se basent essentiellement sur des systèmes dirigés par les données. Cette approche révèle des résultats intéressants mais nécessite l'utilisation d'un grand nombre de données d'apprentissage dans la langue cible. Dans le cas du français, les ensembles de données disponibles sont des corpus annotés dans un premier temps en constituants, puis convertis automatiquement en dépendances. Il en résulte des corpus en dépendances contenant très peu ou pas du tout de dépendances non-projectives. Il n'existe donc pas de corpus en dépendances librement disponible pour le français traitant largement les discontinuités du langage par des dépendances non-projectives.

L'environnement intégré CDG Lab proposé par Béchet *et al.* (2014) permet de développer des corpus en dépendances non-projectifs ainsi que des grammaires catégorielles de dépendances (CDG pour *Categorial Dependency Grammar*) (Dekhtyar et Dikovsky, 2004, 2008) en parallèle. Les CDG permettent de décrire la syntaxe des langues naturelles au moyen de dépendances projectives et non-projectives. Une grammaire du français de grande envergure (Dikovsky, 2011) est utilisée par l'analyseur du CDG Lab pour l'analyse du français. Ce genre d'analyse (à base de règles) génère un nombre exponentiel de possibilités en fonction de la complexité de la phrase à traiter dès lors que la grammaire possède un nombre conséquent de règles. À partir de ce point, il est nécessaire de désambiguïser l'entrée de l'analyseur : fournir plus d'informations sur les phrases, les mots, à l'analyseur pour réduire l'espace de recherche et ainsi le nombre de possibilités en sortie. L'analyseur du CDG Lab propose aux annotateurs de renseigner ces informations en amont de l'analyse

[2]Les dépendances non-projectives peuvent croiser les dépendances projectives dans la structure de dépendances.

en dépendances guidée par la grammaire. Cependant cette tâche est longue et fastidieuse et rend pénible le travail d'annotation des phrases en structures de dépendances. Une première étape de nos travaux est donc de mettre en place des outils d'annotation automatique permettant de réduire la charge de travail des annotateurs et ainsi d'accélérer le processus d'annotation dans le but de construire des corpus en dépendances adaptés au formalisme des CDG i.e. gérant les dépendances projectives et non-projectives dans une même représentation.

Avoir de vastes ensembles de données permet de générer des modèles statistiques couvrant un large panel de phénomènes syntaxiques pour une langue particulière. Les systèmes permettant de créer et d'utiliser ces modèles pour l'analyse en dépendances sont relativement efficaces, en terme de rapidité et de précision. Néanmoins, ces systèmes dirigés par les données sont généralement capables de bien traiter les cas les plus probables mais de passer outre les cas particuliers tel que certains cas non-projectifs spécifiques. Bien qu'il existe différents systèmes gérant les dépendances non-projectives, les scores de ces systèmes sur les dépendances projectives sont largement supérieurs aux scores sur les dépendances non-projectives. Par ailleurs, il apparaît que la recherche simultanée des dépendances projectives et non-projectives influe négativement sur les scores des dépendances projectives. Il semble alors nécessaire d'utiliser les informations des représentations projectives et non-projectives pour améliorer les scores d'analyse en dépendances. Les structures de dépendances produites à partir de l'analyseur du CDG Lab s'avèrent posséder les atouts des deux représentations. En effet, ces structures allient dépendances projectives et non-projectives dans une même représentation, i.e. chaque dépendance non-projective est couplée à une dépendance projective, ce qui permet de pouvoir tirer parti de plus d'informations lors de l'analyse en dépendances. Il est donc intéressant d'étudier des méthodes se basant sur cette représentation pour perfectionner l'analyse en dépendances.

Les travaux de cette thèse vise donc deux objectifs corrélés : diminuer le travail des annotateurs en apportant des solutions pour désambiguïser l'analyse en dépendances guidée par la grammaire dans le but de construire de larges corpus en dépendances non-projectifs, et adapter une méthode d'analyse dirigée par les données à une représentation en dépendances non-projective particulière dans le but de renforcer les performances de l'analyse en dépendances gérant les dépendances non-projectives.

1.3 Contributions

Les premiers travaux présentés dans cette thèse s'intéressent à l'amélioration du processus d'annotation du CDG Lab. En premier lieu, nous avons exposé en quoi la prédiction des étiquettes en dépendances avant analyse est nécessaire pour améliorer l'analyse en dépendances guidée par la grammaire (Lacroix, 2013a,b; Béchet *et al.*, 2014). Nous avons procédé à des expérimentations avec différents outils d'étiquetage (MaxEnt/CRF). Puis nous avons mis en place (Lacroix *et al.*, 2014) un processus complet d'annotation automatique comprenant la segmentation des phrases en mots, l'étiquetage grammatical de ces mots et leur étiquetage syntaxique. Nous avons alors intégré ces différentes étapes à l'environnement intégré CDG Lab et analysé les différentes difficultés qui en découlent (Lacroix et Béchet, 2014b).

La seconde contribution importante de cette thèse concerne l'adaptation d'un analyseur en dépendances par transition à la représentation en dépendances du formalisme des grammaires catégorielles de dépendances. Une première étape consistait à tester (Karlov et Lacroix, 2012) un système d'analyse par transition étendu à la gestion des dépendances non-projectives et des ancres (dépendances projectives) sur un corpus en dépendances du français dont les annotations suivent le formalisme des CDG. Ensuite nous avons proposé une méthode d'analyse par transition séparant les étapes de prédiction des dépendances projectives et des dépendances non-projectives (Lacroix

et Béchet, 2014a). La méthode a été testée au préalable sur le corpus en dépendances du français dont nous disposons. Puis nous avons évalué cette méthode sur des corpus en langue étrangère en procédant auparavant à une conversion de ces corpus pour les adapter à la représentation en dépendances projective/non-projective.

1.4 Plan de la thèse

La première partie de cette thèse expose l'état de l'art de l'analyse en dépendances en 3 chapitres distincts. Dans un premier temps (chapitre 2), nous faisons état des différentes représentations en dépendances existantes, des différentes méthodes d'analyse en dépendances et des corpus disponibles. Dans un second temps (chapitre 3), nous parcourrons l'état de l'art des différent(e)s outils/méthodes dirigé(e)s par les données et utilisé(e)s dans le domaine de l'analyse en dépendances que ce soit lors de pré-traitements ou de l'analyse en elle-même. Puis, nous présentons (chapitre 4) le formalisme particulier que sont les grammaires catégorielles de dépendances, dont la représentation et l'utilisation sont un fil conducteur tout au long de cette thèse.

La seconde partie de la thèse expose (chapitre 5) les différents travaux effectués dans le cadre de l'intégration d'un processus d'annotation automatique dans le mécanisme d'annotation du CDG Lab, pour la construction de corpus en dépendances.

La troisième partie de la thèse présente l'adaptation d'un analyseur par transition à la représentation en dépendances induites par les CDG (chapitre 6) puis la séparation des étapes de prédiction des dépendances projectives et non-projectives lors de l'analyse par transition (chapitre 7).

I

État de l'art

2

Les dépendances : représentations, analyses et corpus

2.1 Introduction

L'analyse en dépendances s'inscrit dans le domaine plus général de l'analyse syntaxique. L'analyse en dépendances découle de théories syntaxiques privilégiant la construction de liens syntaxiques binaires entre les mots lors de l'analyse d'une phrase plutôt que le regroupement des mots en syntagmes (analyse en constituants). Procéder à une analyse en dépendances pour une phrase donnée consiste donc à établir les liens syntaxiques (dépendances) existants entre les mots de la phrase, tels que par exemple les relations sujet-verbe ou verbe-objet, dans le but de construire une structure syntaxique complète pour cette phrase. Historiquement, la représentation en dépendances est tout d'abord employée pour des langues dont l'ordre des mots dans la phrase est flexible (e.g. le russe, l'allemand). Néanmoins, l'analyse en dépendances est aujourd'hui également appliquée à diverses langues dont l'ordre des mots est non-nécessairement flexible. En effet, la représentation en dépendances apporte d'autres avantages que la seule flexibilité des structures. À partir de la représentation en dépendances d'une phrase donnée il est possible d'en extraire facilement des relations prédicat-arguments ce qui peut-être en outre particulièrement avantageux, par exemple, pour l'étude des systèmes de question-réponse ou pour les méthodes d'extraction d'informations.

Dans ce chapitre, nous abordons dans un premier temps le problème de la représentation en dépendances. En effet, les questions liées à l'analyse en dépendances ne sont pas seulement restreintes au simple fait de proposer des méthodes/outils permettant de procéder à des analyses. Tout d'abord se pose la question de ce que l'on souhaite produire, en terme de structure syntaxique. Les représentations en dépendances sont multiples et n'accordent pas toutes la même importance aux différentes informations qui peuvent être intégrées dans une représentation syntaxique. Les formalismes sur lesquels reposent ces représentations sont influencés par différentes théories. Les représentations qui en résultent répondent à différents enjeux : linguistiques et/ou applicatifs. L'objectif peut être d'une part d'atteindre un niveau de représentation syntaxique très précis des langues naturelles. D'autre part, l'objectif peut être de savoir quelles informations privilégier dans la représentation pour que l'analyse en dépendances soit techniquement plus efficace (i.e. moins

ambiguë, plus rapide). Les réponses à ces questions donnent naissance à des représentations utilisant différentes structures de données (e.g. arbres, graphes) et garantissant différentes propriétés (e.g. projectivité, non-projectivité).

L'analyse en dépendances est aujourd'hui fréquemment intégrée dans des procédures du traitement des langues naturelles. En ce sens, ce qui est attendu des systèmes d'analyse en dépendances est une haute efficacité en terme de rapidité/complexité et de précision, et la capacité à produire des structures assez informatives pour représenter au mieux la syntaxe des langues naturelles. Un large choix de méthodes et d'améliorations ont été proposées au fur et à mesure des années pour assurer ces objectifs, l'importance donnée à chacun de ces objectifs variant fortement selon l'orientation des travaux. Les méthodes d'analyse en dépendances peuvent se baser sur des approches symboliques, stochastiques ou alliant les deux. Nous présentons, dans la seconde section de ce chapitre, les fondements des principaux algorithmes et méthodes utilisés dans le domaine ainsi que des analyseurs librement disponibles.

En outre, pour mesurer l'efficacité des analyseurs en dépendances il est nécessaire de tester ceux-ci sur des données suffisantes quantitativement et qualitativement. Pour l'analyse en dépendances, une première approche est de partir des données disponibles pour bâtir une méthode permettant de les traiter telles qu'elles sont. Une seconde approche est de convertir les données disponibles pour faire apparaître les informations qui nous intéressent. La dernière solution est de construire "soi-même" un ensemble de données satisfaisant les caractéristiques que l'on souhaite mettre en avant. Nous présentons donc dans la troisième partie de ce chapitre les corpus en dépendances librement disponibles pour le français et pour d'autres langues ainsi que les représentations qu'ils utilisent et détaillons les processus d'annotation ou de conversion employés pour construire ces corpus.

2.2 Les représentations en dépendances

Il est important de différencier deux aspects distincts mais parfois confondus des représentations en dépendances : l'aspect linguistique et l'aspect informatique. Les premières théories syntaxiques mettant en avant l'usage des dépendances proposaient des représentations graphiques motivées par des principes linguistiques. D'un point de vue informatique il est nécessaire de faire de ces représentations des objets formels. Les descriptions formelles de ces représentations s'inscrivent alors dans le contexte de théories telles que la théorie des graphes et amènent à considérer différentes propriétés sur les objets créés utilisables dans le cadre de la mise en œuvre de méthodes d'analyse en dépendances. Les représentations en dépendances ont pris différentes formes, d'un point de vue graphique aussi bien que d'un point de vue formel, au cours de l'évolution des théories syntaxiques. Néanmoins, par abus de langage le terme **arbre de dépendances** a été et est toujours couramment employé pour désigner tout ensemble de dépendances connexes reliant syntaxiquement les mots d'une phrase donnée, bien que ces représentations ne soient pas toujours formellement des arbres.

Dans cette section nous présentons tout d'abord les travaux qui sont à l'origine des études sur les représentations syntaxiques en dépendances à travers une petite histoire de la représentation en dépendances. Puis nous définissons ensuite les objets formels qui sont aujourd'hui couramment employés dans le domaine de l'analyse en dépendances et nous présentons certaines variantes qui ont été proposées dans le but d'apporter des informations supplémentaires dans les représentations en dépendances.

2.2.1 Une petite histoire de la représentation en dépendances

Dans le domaine de la syntaxe, les travaux de Tesnière (1959) sont ceux qui présentent une première approche de la représentation en dépendances. Il y proposait la notion de « stemma » s'apparentant aux arbres de dépendances couramment exploités aujourd'hui. Dans ces premières représentations des arbres de dépendances, le mot racine de la structure (il est le seul mot à n'avoir pas de gouverneur) est le mot le plus haut de l'arbre et est celui dont dépendent (directement ou indirectement) tous les autres mots de la phrase. Un gouverneur y est toujours représenté au dessus de son subordonné. Les arcs ne sont dans ce cas pas graphiquement orientés mais l'orientation est induite par cette hiérarchie. Un exemple d'arbre de dépendances vu par Tesnière est donné par la figure 2.1. Il est à noter que l'ordre des mots n'y est pas conservé. Il n'y a alors pas de croisements entre les branches de l'arbre.

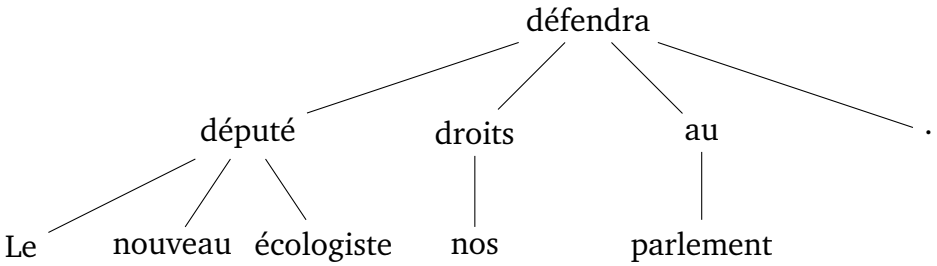


FIGURE 2.1 – Arbre de dépendances pour la phrase « Le nouveau député écologiste défendra nos droits au parlement. » selon les travaux de Tesnière (1959). Le verbe « défendra » est la racine de la phrase.

Tesnière propose également qu'un mot puisse avoir deux gouverneurs si le mot y est syntaxiquement lié à parts égales comme dans le cas d'une coordination. On se trouve dans ce cas avec des objets qui sont plus que des arbres. Dans l'exemple présenté par la figure 2.2 on est dans le cas d'un graphe non-orienté. Une telle représentation peut alors engendrer des cycles dans la structure.

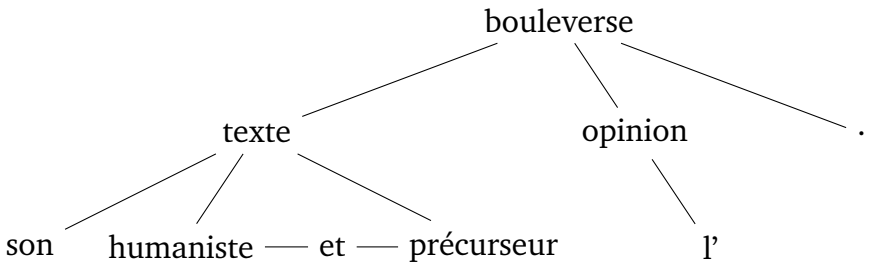


FIGURE 2.2 – Graphe de dépendances pour la phrase « Son texte humaniste et précurseur bouleverse l'opinion. ». Les adjectifs « humaniste » et « précurseur » s'appliquent tous les deux au mot « texte » et sont liés par la coordination « et ».

Aujourd'hui les représentations en dépendances couramment exploitées dans le domaine de l'analyse en dépendances sont appelées des **représentations syntaxiques de surface**. Le terme de représentation syntaxique de surface provient des travaux de Mel'cuk (1988) dans lesquels il propose, à travers la théorie Sens-Texte, plusieurs niveaux d'analyse linguistiques (phonologie, morphologie, syntaxe et sémantique). Certains niveaux linguistiques sont eux-même divisés en deux niveaux : une représentation de surface et une représentation profonde. Ces différents niveaux, et les liens qu'ils partagent, sont exposés dans la figure 2.3. L'idée de la théorie Sens-Texte

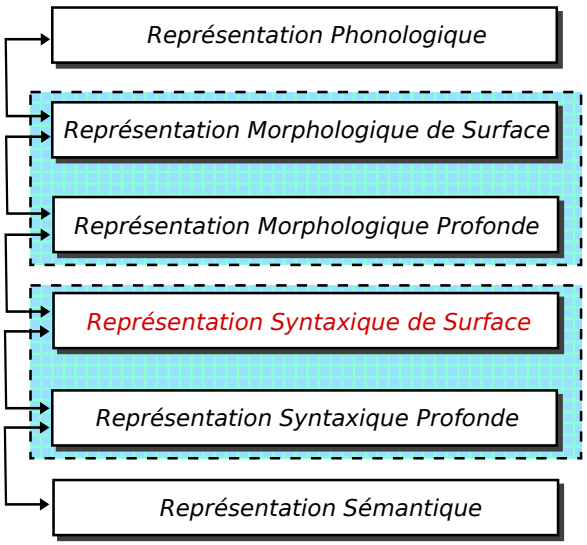
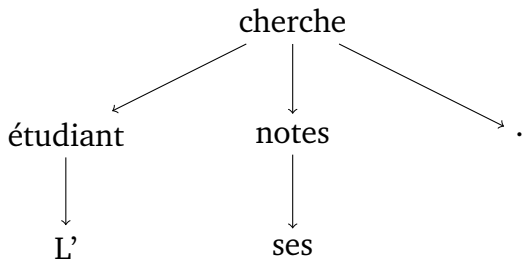


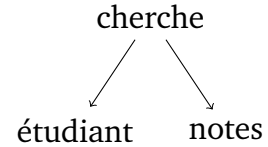
FIGURE 2.3 – Représentation des niveaux d'analyse linguistiques d'après la théorie Sens-Texte de Mel'cuk (1988).

est d'établir des règles linguistiques permettant de passer d'un niveau de représentation à un autre (lorsque ceux-ci sont directement liés). Dans chaque représentation, les dépendances jouent un rôle spécifique. Les dépendances permettent d'établir des relations binaires entre les mots des phrases et donc de spécifier les fonctions que les mots ont les uns envers les autres. La direction des dépendances dans une phrase détermine une hiérarchie entre les mots. D'après Mel'cuk, la représentation de la syntaxe des phrases par les dépendances est plus naturelle que la représentation en syntagme pour les langues dont l'ordre des mots est relativement flexible, tel que le russe. Par ailleurs, un autre avantage de l'utilisation des dépendances et de pouvoir en déduire plus aisément des structures prédicat-argument et donc une modélisation sémantique des phrases.

À travers le domaine de l'analyse en dépendances, nous nous intéressons particulièrement au niveau de représentation syntaxique de surface. Notons que cette représentation syntaxique de surface partage un lien fort avec la représentation morphologique profonde tandis que la représentation syntaxique profonde est en corrélation avec la représentation sémantique. La représentation syntaxique profonde ne représente que les mots porteurs de sens dans la phrase d'où son rapport avec la représentation sémantique. Selon Mel'cuk, la représentation profonde a pour but de s'affranchir de la langue et donc de proposer un schéma de représentation universel des dépendances. Alors que la représentation syntaxique de surface conserve tous les mots de la phrase (y compris les mots outils). Elle dépend fortement de la syntaxe de la langue étudiée. Ces deux représentations



(a) Représentation syntaxique de surface



(b) Représentation syntaxique profonde

FIGURE 2.4 – Représentations en dépendances de surface et profonde pour la phrase « L'étudiant cherche ses notes. ».

sont illustrées dans la figure 2.4. On notera que les dépendances sont ici des arcs orientés et que la linéarité de la phrase n'est pas préservée contrairement aux représentations de surface standards actuelles dans lesquelles l'ordre des mots est conservé.

Dans ces travaux, Mel'cuk parle d'arbre de dépendances lorsqu'il met ceux-ci en opposition avec les arbres en constituants mais il emploie fréquemment le terme de **graphe** ou de **structure** pour désigner les représentations en dépendances des phrases qu'il propose en exemple.

2.2.2 Dépendances et arbres : définitions et notations

Il n'existe pas une seule manière de représenter l'ensemble des dépendances syntaxiques d'une phrase donnée mais toutes les représentations en dépendances ont en commun l'usage des dépendances syntaxiques. Dans cette thèse, nous considérons que toute dépendance est orientée, qu'elle décrit donc une relation syntaxique non-symétrique entre deux mots.

Définition 1 *Dépendance (ou dépendance syntaxique)*
*Une dépendance est définie par un triplet (i, d, j) où i est l'indice du mot **gouverneur** de la dépendance (également appelé **tête**), j est l'indice du mot **subordonné** de la dépendance (également appelé **dépendant**) et d est l'étiquette/le nom de la dépendance. On dit que le mot w_j dépend du mot w_i.*

L'étiquette (syntaxique) d'une dépendance décrit la relation/le rôle que les mots gouverneur et subordonné ont l'un pour l'autre. Notons que certaines représentations ne font usage que de dépendances non-étiquetées, i.e. dans ces représentations seules les dépendances syntaxiques sont représentées et non les rôles syntaxiques (i.e. les étiquettes des dépendances) associés à ces dépendances. En outre, une dépendance définie une relation de dominance qui s'établit entre le mot gouverneur et le mot subordonné tel que ce dernier est dominé par le premier. On note $w_i \rightarrow w_j$ (w_i *domine* w_j) pour toute dépendance de la forme (i, d, j). Il s'agit en ce cas d'une dominance directe. Mais une relation de dominance s'établit également entre deux mots reliés indirectement (subordonnés de subordonnés). Dans ce cas, on note $w_i \overset{*}{\rightarrow} w_j$, i.e. il existe un chemin reliant w_i à w_j tel que $w_i \rightarrow ... \rightarrow w_j$. Les propriétés de la relation de dominance sont :

- la réflexivité : $w_i \overset{*}{\rightarrow} w_i$;
- et la transitivité : si $w_i \overset{*}{\rightarrow} w_k$ et $w_k \overset{*}{\rightarrow} w_j$ alors $w_i \overset{*}{\rightarrow} w_j$.

Dans un arbre de dépendance, un mot domine de manière directe ses subordonnés, mais il domine également les subordonnés de ses subordonnés et lui-même. Ainsi la racine domine tous les mots d'une phrase et n'a, elle même, pas de gouverneur.

Définition 2 *Arbre de dépendances*
Un arbre de dépendances pour une phrase $W = w_1...w_n$ donnée est un arbre (i.e. un graphe acyclique et connexe) $G = \{V,A\}$ dont chaque nœud $w_i \in V$ (pour $i \in \{1,...,n\}$) correspond à un mot de la phrase et dont chaque arc représente une dépendance $(i,d,j) \in A$ (pour $i,j \in \{1,...,n\}$ et $i \neq j$) entre deux mots distincts. Chaque mot accepte un et un seul gouverneur (excepté la racine qui n'a pas de gouverneur).

Dans les représentations en dépendances non-linéaires (i.e. dans lesquelles l'ordre des mots n'est pas conservé) telles que celles proposées par Tesnière (sous-section 2.2.1) , il n'y a pas de croisements entre les arcs des arbres de dépendances. Aujourd'hui les arbres de dépendances sont fréquemment représentés à travers des représentations en dépendances de surface linéaires, dans

lesquelles l'ordre des mots est conservé. Une telle représentation correspond à la projection des arbres de dépendances vus précédemment. Les arcs sont nécessairement orientés pour préserver la relation de dominance qui spécifie un gouverneur et un subordonné pour chaque dépendance. Le mot racine de la phrase n'ayant pas de gouverneur reçoit une dépendance entrante n'ayant pas d'origine. De plus, la projection d'un arbre peut engendrer des croisements entre les dépendances dans le cas d'une discontinuité de la langue. Cette représentation linéaire et plate de l'arbre de dépendances permet donc de mieux observer sa projectivité ou sa non-projectivité (voir les définitions 4 et 6). Un exemple d'arbre de dépendances représenté linéairement est donné par la figure 2.5.

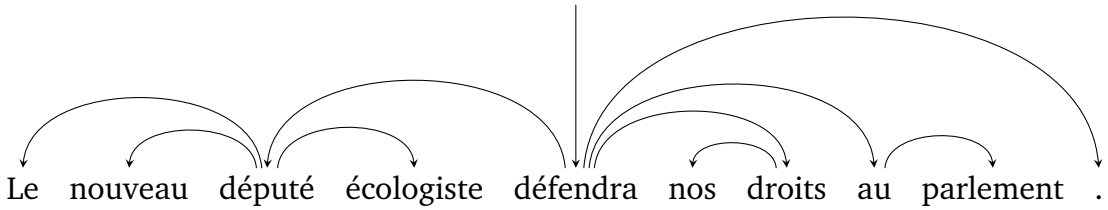


FIGURE 2.5 – Projection de l'arbre de dépendances vu dans la figure 2.1 pour la phrase « Le nouveau député écologiste défendra nos droits au parlement. ».

En outre, on appelle projection d'un mot l'ensemble des mots dominés par ce mot : comprenant lui-même, ses subordonnés, les subordonnés de ses subordonnés, etc. Par exemple, la projection du mot « député » dans la figure 2.5 est le sous-arbre engendré à partir de ce mot, i.e. comprenant les mots « Le », « nouveau », « député » et « européen ». L'ensemble de ces mots forment dans cet exemple un intervalle continu (un sous-segment strict de la phrase donnée). La projection du mot racine d'un arbre est la phrase entière. Une définition formelle de la projection d'un mot est donnée par la définition 3. Elle permet de préciser la propriété de **projectivité** d'un arbre de dépendances (définition 4).

Définition 3 *Projection d'un mot*
La projection d'un mot w appartenant à une phrase W est $P(w) = \{w' \in W \mid w \xrightarrow{*} w'\}$.

Définition 4 *Arbre projectif*
Un arbre de dépendances pour une phrase W est projectif si chaque mot $w \in W$ a une projection $P(w)$ définie sur un intervalle continu de W.

En d'autres termes, un arbre est projectif si pour chaque mot w d'une phrase W tous les mots qui se trouvent entre w et ses subordonnés sont aussi dominés par w. Visuellement, la projectivité d'un arbre de dépendances se remarque par l'absence de croisements entre les dépendances dans la représentation linéaire de cet arbre. Au contraire, les croisements déterminent la **non-projectivité** d'un arbre. Les propriétés de non-projectivité des dépendances et des arbres de dépendances sont décrites par les définitions 5 et 6. La figure 2.6 présente un arbre non-projectif.

Définition 5 *Dépendance non-projective*
Une dépendance (i, d, j) est non-projective si la projection $P(w_i)$ du gouverneur w_i sur l'intervalle $[i,j]$ (pour une dépendance droite, $[j,i]$ pour une dépendance gauche) n'est pas continu.

Définition 6 *Arbre non-projectif*
Un arbre de dépendances pour une phrase W est non-projectif si au moins un mot $w \in W$ a une projection $P(w)$ définie sur un intervalle discontinu de W.

La définition de la non-projectivité d'une dépendance (i, d, j) signifie que si au moins un des mots situés entre le gouverneur w_i de la dépendance et son subordonné direct w_j n'est pas dominé (i.e. ne dépend pas indirectement ou directement) du gouverneur w_i alors la dépendance est non-projective. Dans la figure 2.6, le mot « fragile » compris entre « plus » et « que » n'est pas dominé par le gouverneur « plus », la dépendance (relation de comparaison) liant « plus » et « que » est donc une dépendance non-projective. L'arbre est non-projectif car au moins une de ses dépendances est non-projective.

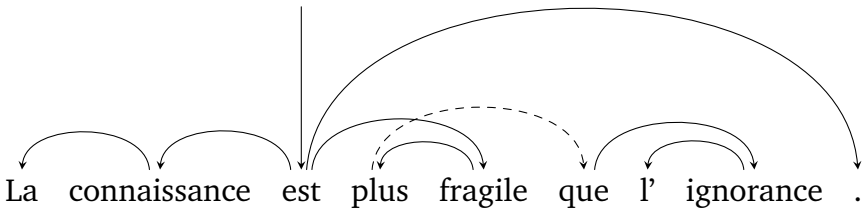


FIGURE 2.6 – Arbre de dépendances non-projectif pour la phrase « La connaissance est plus fragile que l'ignorance ».

La représentation linéaire des arbres de dépendances est aujourd'hui un format standard de représentation en dépendances, dans le cas projectif comme dans le cas non-projectif. Le choix d'une représentation projective ou non-projective découle de différentes raisons, linguistiques ou informatiques. En effet, la syntaxe d'une langue peut ou pas nécessiter l'emploi des dépendances non-projectives (certaines langues n'incluent pas ou peu de cas de discontinuité comme par exemple le chinois (Liu et Huang, 2006)) mais cette non-projectivité peut également être ignorée pour des langues qui admettent pourtant des cas discontinus. Ce dernier point est approfondi dans la section 2.4 présentant les corpus en dépendances.

Dans le cas du français, certains phénomènes linguistiques sont communément reconnus comme des cas de discontinuité de la langue. La figure 2.6 propose un exemple dans lequel la relation de comparaison admise entre les mots « plus » et « que » induit une dépendance non-projective. L'utilisation des clitiques dans la langue française induit également des discontinuités comme par exemple l'emploi du clitique « en » dans « J'ai choisi le modèle, et j'en ai fait une sculpture. » où « en » est dépendant de « sculpture ». Le clitique « y » dans « Y avez-vous pensé ? » est également discontinu puisqu'il dépend de « pensé ». Les objets extraits de leur groupe verbal peuvent induire des dépendances non-projectives (e.g. l'objet « idée » du verbe « eue » dans la phrase « Mais quelle drôle d'idée elle a eue ! ») ainsi que lorsqu'un modificateur est distant (e.g. « Elle s'est arrêtée, prise de crainte » où « prise » modifie « Elle »). En outre, selon les schémas d'annotation, des dépendances non-projectives peuvent également apparaître par exemple lors de l'introduction d'éléments de négation (e.g. « Jamais mes hommes ne feraient ça. » où « ne » dépend de « Jamais »), lorsqu'un sujet est réintroduit par un pronom démonstratif (e.g. « Les loisirs, à la montagne, ce n'est pas de tout repos » où « loisirs » dépend de « ce ») ou précisé ultérieurement (« Elle est grande, sa fille ! » où « fille » fait référence à « Elle »). La figure 2.7 présente un arbre de dépendance dans lequel trois cas discontinus sont pris en compte :

- le clitique « l' » fait référence à l'objet de « trouvée » ;
- le mot « table » fait lui-même référence à l'objet introduit par le clitique « l' » ;
- et « en » est rattaché à « table » en tant qu'attribut permettant d'introduire une description.

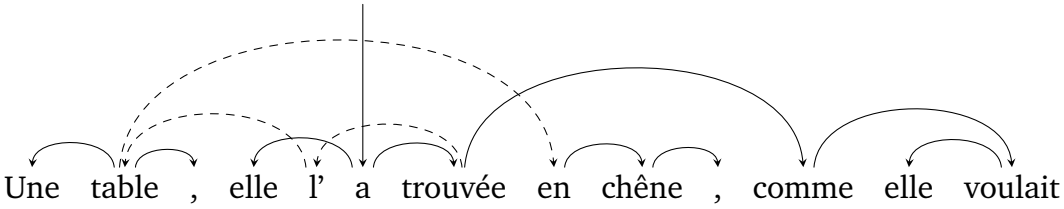


FIGURE 2.7 – Arbre de dépendances non-projectif pour la phrase « Une table, elle l'a trouvée en chêne, comme elle voulait ».

2.2.3 Graphes et variantes

Les arbres de dépendances sont des graphes particuliers, ils sont acycliques, connexes et orientés. L'emploi de ces arbres pour la représentation en dépendances peut être vu comme une contrainte restreignant l'expressivité de certaines constructions syntaxiques. Il est alors fréquent que des théories syntaxiques se démettent de certaines des propriétés des arbres de dépendances pour autoriser l'ajout d'informations par de nouvelles dépendances. Dans ce cas, on travaille alors généralement avec des graphes de dépendances.

Il existe par exemple différentes représentations en dépendances admettant qu'un nœud ait plus d'un gouverneur. Cet ajout de liens peut être justifié par différents aspects syntaxiques. Nous avons vu que Tesnière (1959) (sous-section 2.2.1) proposait qu'un mot puisse avoir deux gouverneurs si le mot y est syntaxiquement lié à parts égales, menant les représentations en dépendances à recourir à des graphes non-orientés. Cependant, la majorité des représentations exploitées aujourd'hui sont des graphes orientés. L'orientation des arcs détermine généralement l'absence de cycles dans les graphes de dépendances. On obtient dans ce cas des représentations utilisant les DAGs (*Directed Acyclic Graph*) ou graphes orientés acycliques.

Par exemple, le formalisme des grammaires catégorielles de dépendances (Dekhtyar et Dikovsky, 2004) amène à produire des DAGs permettant de prendre en compte et de mettre en évidence les discontinuités des langues. Il s'agit d'une représentation linéaire de surface dans laquelle les mots sont reliés par différents types de dépendances. Chaque dépendance non-projective dans la structure est couplée à une dépendance projective. Par conséquent, les mots créant des discontinuités dans la phrase possèdent deux gouverneurs et non un seul. Dans le cadre de ces travaux, la représentation en dépendances employée est désignée par le terme **structure de dépendances**. Une structure de dépendances illustrant cette représentation est présentée dans la figure 2.8. Cette

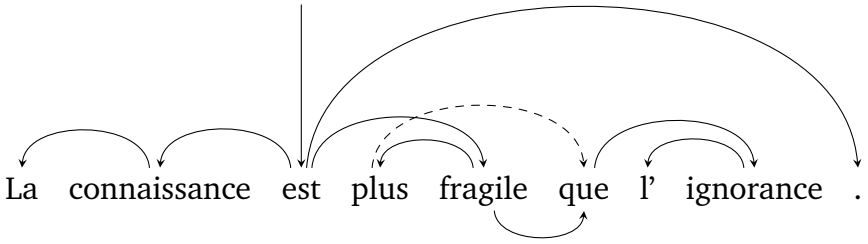


FIGURE 2.8 – Structure de dépendances non-projective pour la phrase « La connaissance est plus fragile que l'ignorance » conforme à la représentation en dépendances induite par les grammaires catégorielles de dépendances.

représentation en dépendances est au cœur de nos travaux, elle fait donc l'objet de la section 4.2 du chapitre 4.

Toutefois, il existe des représentations basées sur les graphes de dépendances (orientés) acceptant les cycles. Par exemple, l'une des représentations en dépendances de Stanford (de Marneffe et Manning, 2008), dite « *collapsed* », admet les cycles entre deux noeuds seulement. En tout, 5 représentations en dépendances de Stanford sont proposées. La représentation de base est un format classique d'arbre projectif tandis que la représentation de base étendue accepte les dépendances croisées. En parallèle, la représentation « *collapsed* » propose de relier de manière directe les mots porteurs de sens en épurant la structure de certains mots outils i.e. en ajoutant des informations sur ces mots outils dans les étiquettes des dépendances. Cette représentation s'apparente alors nettement à une représentation profonde, plus sémantique. Les deux dernières représentations sont des variantes de la représentation « *collapsed* » traitant le cas particulier des conjonctions pour l'une et restreignant la structure à un arbre pour l'autre.

Le phénomène de la coordination pose souvent problème dans les représentations. Kahane (1997) propose, par ailleurs, une représentation basée sur les *arbres à bulles*. Ces arbres permettent de rassembler des mots en bulles lorsqu'ils partagent un même lien syntaxique avec un autre mot comme dans les divers cas de coordination.

Au cours du temps, les études sur la syntaxe des langues ont donc donné lieu à différentes représentations en dépendances. Dans le cadre de nos travaux, nous employons principalement la représentation induite par les grammaires catégorielles de dépendances. Dans cette thèse, nous désignerons alors communément les représentations en dépendances par le terme **structure de dépendances**.

2.3 Les méthodes d'analyse syntaxique et l'analyse en dépendances

Le but de l'analyse en dépendances est de construire, pour une phrase donnée, la structure de dépendances correcte révélant la syntaxe de la phrase. Néanmoins, les systèmes d'analyse en dépendances n'étant pas parfaits, la résolution de ce problème s'accompagne de sous-objectifs complémentaires que sont la recherche de la précision et de la rapidité. Certains systèmes sont donc plus précis et d'autres plus rapides selon l'axe privilégié par les travaux. Nous présentons dans cette section les différentes méthodes permettant de procéder à l'analyse en dépendances des langues naturelles selon leur axe de recherche.

Les méthodes d'analyse en dépendances peuvent être basées sur des modèles symboliques et/ou probabilistes. Couramment, on opposera ces deux méthodes en tant que méthodes **basées sur les grammaires** (*grammar-based*) ou **dirigées par les données** (*data-driven*). Cependant, il n'est pas inhabituel de voir proposer des travaux alliant les avantages des deux méthodes. Les méthodes basées sur les grammaires sont assez classiques dans le sens où elles utilisent principalement les mêmes algorithmes d'analyse plus ou moins étendus. La différence s'observe au niveau des différents formalismes grammaticaux employés pour décrire la syntaxe d'un langage. D'un autre côté, les méthodes dirigées par les données se différencient les unes des autres par les systèmes qu'elles emploient mais se basent toutes sur un apprentissage automatique supervisé à partir d'ensembles de données correctement annotés.

2.3.1 Analyses syntaxiques basées sur les grammaires

Les méthodes d'analyse syntaxique basées sur les grammaires sont des méthodes guidées par les règles de grammaires formelles. Nous avons précisé que le but de l'analyse syntaxique est de produire une structure syntaxique correcte pour une phrase donnée. Dans le cadre des analyses basées sur les grammaires, il s'agit communément de couvrir l'ensemble des analyses possibles pour la phrase donnée, puis d'en faire ressortir une parmi celles-ci (i.e. de trier les analyses). Les méthodes permettant de trier les analyses produites sont parfois intégrées aux algorithmes d'analyse ou parfois réalisées ultérieurement mais tirent généralement profit de principes probabilistes. Nous présentons plus précisément ces méthodes à travers les sous-sections 3.3.2 et 3.4.1 du chapitre 3. Nous nous restreignons ici à présenter les algorithmes de base des méthodes basées sur les grammaires.

Le principe d'une analyse basée sur les grammaires est le même dans le cas d'une analyse en constituants ou dans le cas d'une analyse en dépendances. Les mêmes algorithmes sont à l'origine de ces méthodes. Les variantes et les formalismes grammaticaux proposés pour l'analyse permettent alors de différencier les théories. De plus, ces algorithmes sont à l'origine de la seconde appellation de ce type d'analyse : les analyses tabulaires (*chart parsing* en anglais). Il s'agit principalement d'algorithmes permettant d'utiliser des grammaires hors-contexte pour effectuer des analyses en un temps polynomial. Ces grammaires présentent un formalisme général de base permettant de décrire les langages algébriques et donc adaptés aux langues naturelles. En outre, les formalismes grammaticaux particuliers que nous présentons plus précisément ensuite restent équivalents avec les grammaires hors-contexte (CFGs pour *Context Free Grammars*). Les CFGs sont décrites dans la définition 7. L'ensemble des symboles non-terminaux N, décrit dans cette définition, inclue généralement des fonctions syntaxiques et potentiellement des catégories grammaticales (parties du discours).

Définition 7 *Grammaire hors-contexte*
Une grammaire hors-contexte est définie par un quadruplet $G = \langle N, \Sigma, \Pi, S \rangle$ *où :*

- N *est en ensemble fini de symboles non-terminaux ;*
- Σ *est un ensemble fini de symboles terminaux i.e. les mots de la langue cible ;*
- Π *est un ensemble de règles de la forme* $X \rightarrow \alpha$ *où* $X \in N$ *et* $\alpha \in (\Sigma \cup N)^*$ *;*
- S *est un symbole de départ i.e. l'axiome.*

Les algorithmes de référence pour les méthodes d'analyse tabulaire sont des algorithmes tels que l'algorithme CKY de Cocke, Kasami et Younger (Younger, 1967) ou l'algorithme d'Earley (1970). Un algorithme d'analyse tabulaire (mais n'utilisant pas de grammaire) plus récent et spécialement adapté à l'analyse en dépendances est celui d'Eisner (1996).

L'algorithme de Cocke, Kasami et Younger est un algorithme d'analyse syntaxique standard en programmation dynamique. La complexité de l'algorithme est au pire de $O(n^3 \mid G \mid)$ avec G une grammaire hors-contexte en forme normale de Chomsky (1959) et n la taille de la phrase à analyser (i.e. le nombre de mots). Il s'agit d'un algorithme qui part des mots de la phrase pour appliquer les règles de production adéquates jusqu'à trouver une bonne analyse. Le principe de l'algorithme est décrit par l'Algorithme 1.

Algorithme 1 : CKY

Données : $W = w_1...w_n$ une phrase,
$G = \langle N,\Sigma,\Pi,R_s\rangle$ une grammaire avec $N = \{R_1,...,R_r\}$ l'ensemble des non-terminaux,
$P[n,n,r]$ un tableau de booléen initialisé à faux.

```
pour i ← 1 à n faire
    pour chaque règle de production de la forme R_k → w_i faire
        P[i,1,k] ← vrai
pour i ← 2 à n faire
    pour j ← 1 à n − i + 1 faire
        pour k ← 1 à i − 1 faire
            pour chaque règle de production de la forme R_A → R_B R_C faire
                si P[j,k,B] et P[j + k,i − k,C] alors
                    P[j,i,A] ← vrai
si ∀x ∈ R_s, ∃P[1,n,x] == vrai alors
    W appartient au langage
sinon
    W n'appartient pas au langage
```

L'algorithme CKY, et des variantes de cet algorithme, sont toujours utilisés dans le domaine de l'analyse syntaxique que ce soit pour l'analyse en constituants ou l'analyse en dépendances. Une version adaptée aux grammaires hors-contexte probabilistes (PCFG pour *Probabilistic Context-Free Grammar*) fut aussi proposée. Il est possible de trouver une description de cette version de l'algorithme dans l'ouvrage de Jurafsky et Martin (2000).

L'algorithme de Earley utilise la programmation dynamique pour effectuer une analyse d'une complexité au pire en $O(n^3)$ où n est la taille de la phrase à analyser. Le principe de l'algorithme est de parcourir un tableau de taille $n+1$: chaque entrée comprend une liste d'états à appliquer en fonction du mot à lire dans la phrase. Les états correspondent à des « états de lecture » d'une règle de la grammaire. Formellement, un état est un couple $\langle R,i\rangle$ où R est une règle de production de la grammaire dont la position de lecture dans la production est indiquée et i est la position de lecture dans la phrase. R est décrit par une formule de la forme $X \rightarrow \alpha \bullet \beta$ où le point « $\bullet$ » correspond à la position de lecture dans la production et α, $\beta \in \{N \cup \Sigma\}^*$. L'algorithme parcours donc la liste des états de chaque entrée du tableau et applique pour chaque état l'une des opérations suivantes sous des conditions particulières :

- **Prédiction** (*Predictor*), cette opération est appliquée lorsque R est de la forme $X \rightarrow \alpha \bullet B\beta$ où B est un symbole non-terminal représentant une fonction syntaxique. Dans ce cas, de nouveaux états de la forme $B \rightarrow \bullet\gamma$ sont générés à partir des règles de la grammaire ;

- **Lecture** (*Scanner*), cette opération est appliquée lorsque R est de la forme $X \rightarrow \alpha \bullet B\beta$ où B est un symbole non-terminal représentant une catégorie grammaticale. Dans ce cas, si le mot courant w appartient à la catégorie B, deux nouveaux états de la forme $B \rightarrow w\bullet$ et $X \rightarrow \alpha B \bullet \beta$ sont ajoutés à la liste de l'entrée suivante dans le tableau ;

- **Complétion** (*Completer*), cette opération est appliquée lorsque R est de la forme $X \rightarrow \gamma\bullet$. Dans ce cas, pour chaque état de la forme $Y \rightarrow \alpha \bullet X\beta$, un état de la forme $Y \rightarrow \alpha X \bullet \beta$ est ajouté.

Ces méthodes d'analyse classiques ont été largement étudiées et furent exploitées et modifiées de nombreuses fois dans le but d'adapter les algorithmes aux spécificités des grammaires conçues pour l'analyse en constituants ou en dépendances. Le domaine de l'analyse syntaxique compte donc beaucoup de grammaires particulières dérivées de formalismes grammaticaux tout aussi variés. Parmi les premiers travaux sur les grammaires de dépendances on peut citer ceux de Gaifman (1965) et Hays (1965). Ces grammaires sont très proches des CFGs (elles sont faiblement équivalentes, dans le cas projectif, avec les CFGs (Kübler *et al.*, 2009)) et peuvent donc également exploiter le potentiel des algorithmes d'analyse décrits précédemment. Puis, parmi les formalismes présentés et repris dans divers travaux, on peut citer les grammaires d'arbres adjoints (TAGs pour *Tree-Adjoining Grammars*) (Joshi *et al.*, 1975), les HPSGs (*Head-Driven Phrase Structure Grammars*) (Pollard et Sag, 1987), la grammaire de mots ou *Word Grammar* de Hudson (1990), les LFGs (*Lexical Functional Grammars*) (Kaplan et Bresnan, 1995), les grammaires de liens ou *Link Grammars* de Sleator et Temperley (1995), les XDGs ou *Extensible Dependency Grammars* (Debusmann *et al.*, 2004) descendant des TDGs (*Topological Dependency Grammars*) (Duchier et Debusmann, 2001). Nous allons particulièrement nous attacher à décrire les spécificités des TAGs, de la *Word grammar* et de la *Link Grammar* pour leurs similitudes avec les théories syntaxiques en dépendances.

Les TAGs ou grammaires d'arbres adjoints sont une classe de grammaires dont les fondements furent présentés par Joshi *et al.* (1975). Elles furent ensuite spécialisées avec les LTAGs (*Lexicalised Tree-Adjoining Grammar*) (Schabes, 1990). Aujourd'hui la majorité des travaux sur les grammaires d'arbres adjoints font en fait référence aux LTAGs. Les TAGs peuvent être rapprochées des grammaires hors-contexte, cependant ici les règles de la grammaire ne sont pas des règles de production classiques mais des arbres élémentaires auxquels s'appliquent des opérations de substitution et d'adjonction. Les LTAGs sont lexicalisées dans le sens où chaque arbre élémentaire possède au moins une feuille qui est un symbole terminal i.e. un mot. Ainsi, à chaque mot est associé un ensemble d'arbres élémentaires représentant les contextes syntaxiques possibles pour ce mot dans la langue étudiée.

Le principe d'une analyse par TAG est que certains nœuds des arbres élémentaires, n'étant pas des symboles terminaux, soient remplacés par d'autres arbres élémentaires dont la racine est un symbole terminal de même catégorie syntaxique, pour trouver à terme un arbre de dérivation correspondant à une analyse correcte pour une phrase donnée. La substitution consiste plus particulièrement à compléter des arbres en remplaçant des feuilles par de nouveaux arbres élémentaires. Tandis que l'adjonction consiste à remplacer des nœuds internes de l'arbre de dérivation pour ajouter des éléments syntaxiques.

Le liens entre les TAGs et le domaine de l'analyse en dépendances est dû au fait qu'elles permettent également de considérer certains phénomènes syntaxiques discontinus. Les arbres de dérivation peuvent être non-projectifs et assimilés à des arbres en dépendances. Notons, par ailleurs, les travaux de Joshi et Rambow (2003) proposant un formalisme similaire aux TAGs pour l'analyse en dépendances utilisant des arbres élémentaires en dépendances.

La *Word Grammar* décrit une théorie plus qu'une grammaire. Hudson (1990) voit le langage comme un réseau de concepts. Ainsi, la *Word Grammar* rassemble plusieurs niveaux de description du langage parmi lesquels les niveaux morphologique, syntaxique et sémantique. L'idée est alors de construire un réseau reflétant les différents niveaux de relations existants entre les mots d'une langue. Le lien privilégié de la *Word Grammar* est le lien **is-a** qui permet de révéler une hiérarchie entre les mots (e.g. pour spécifier qu'un chat *est un* animal). Ce lien particulier permet de classifier les mots par exemple pour dire qu'un mot est un verbe. On obtient donc une classification grammaticale par le biais d'un tel réseau.

En ce qui concerne la syntaxe, Hudson tient grandement à ce que les structures syntaxiques soient des structures en dépendances et non des structures en constituants puisque le réseau qu'il présente établit des liens entre les mots. Les structures en dépendances de la *Word Grammar* sont alors intégrées dans l'ensemble de la théorie basée sur les réseaux. Les dépendances sont aussi liées par des relations **is-a**. Par exemple, la relation « objet » est une relation « complément ». La finalité de la *Word Grammar* est de pouvoir facilement étudier la sémantique et la syntaxe des langues à travers des réseaux incluant toutes les informations nécessaires dans un format équivalent sur les différents niveaux.

La *Link Grammar* ou *grammaire de lien* est un formalisme qui fut proposé par Sleator et Temperley (1995) et appliqué à l'anglais. L'idée est de connecter les mots des phrases par paires selon certaines conditions de manière à obtenir une structure de liens planaire et connexe. Bien que les propriétés des structures produites soient proches des structures en dépendances projectives, les auteurs ne décrivent pas la *Link Grammar* comme étant une grammaire de dépendances. Les structures de liens sont orientées mais ces liens ne définissent pas une notion de dominance.

Le principe de la grammaire repose sur la notion de connecteur. À chaque mot d'un lexique est associé un connecteur. Un connecteur est une formule composée d'opérateurs and et or, de parenthèses et d'un nom correspondant à une relation syntaxique, couplée à un symbole + ou - indiquant la direction du connecteur (gauche ou droit). Les formules expriment en fait des conditions. Il est alors possible de lier deux mots du lexique ayant des connecteurs compatibles i.e. opposés (+ et -) et satisfaisant les mêmes conditions. L'objectif d'une analyse syntaxique avec la *Link Grammar* est alors de trouver des connections correctes entre les mots tout en respectant les propriétés de planarité et de connectivité. Un algorithme de programmation dynamique basé sur la recherche d'une triangulation optimale dans un polygone convexe fut proposé conjointement pour l'analyse des grammaires de liens en $O(n^3)$. Par ailleurs, de part le fait que chaque mot du lexique dispose d'une définition, la *Link Grammar* fait partie de l'ensemble des grammaires lexicalisées.

Dans le domaine de l'analyse syntaxique basée sur les grammaires, on peut aussi citer le cas des grammaires catégorielles (Bar-Hillel *et al.*, 1964). Ce formalisme particulier a un fort aspect logique. Les grammaires catégorielles de dépendances (Dekhtyar et Dikovsky, 2004; Dikovsky, 2011) sont des grammaires dérivées des grammaires catégorielles s'appliquant à l'analyse en dépendances. Ce formalisme étant à la source de nos travaux, nous présentons plus précisément les grammaires catégorielles classiques, les grammaires catégorielles de dépendances et les applications qui en découlent dans un chapitre spécifique (chapitre 4).

2.3.2 Analyses syntaxiques dirigées par les données

Le principe des méthodes dirigées par les données est d'entraîner, dans un premier temps, des modèles statistiques sur des données correctement annotées et d'utiliser ensuite ces modèles pour

l'analyse de nouvelles données. Ces méthodes comprennent alors deux étapes qui sont : une étape d'apprentissage automatique supervisé et une étape d'analyse syntaxique.

Sommairement, l'étape d'apprentissage supervisé telle qu'utilisée dans les méthodes dirigées par les données consiste à collecter des informations particulières (sous forme de traits) sur des données correctement annotées et à en extraire des probabilités (e.g. probabilité d'une étiquette en fonction de différents critères grammaticaux) dans le but de les exploiter durant l'étape d'analyse syntaxique. Pour les méthodes d'analyse dirigées par les données, il n'est donc pas nécessaire de construire des grammaires puisque ces méthodes tirent profit des outils de l'apprentissage automatique pour apprendre automatiquement des modèles[1]. Si les ensembles de données disponibles sont suffisamment larges et hétérogènes, les modèles appris à partir de ces données sont capables d'être raisonnablement précis. Les outils du domaine de l'apprentissage automatique sont détaillés plus largement dans le chapitre 3.

Dans cette section nous nous concentrons sur les méthodes employées lors de l'étape d'analyse, considérant l'apprentissage comme une étape acquise. D'après Kübler *et al.* (2009), il existe principalement deux méthodes courantes d'analyse en dépendances dirigées par les données. Il s'agit de l'analyse par satisfaction de contraintes (*graph-based*) et de l'analyse par transition (*transition-based*). Nous présentons ci-dessous ces deux méthodes.

• L'analyse par satisfaction de contraintes

L'analyse par satisfaction de contraintes est une méthode d'analyse basée sur les graphes mise en avant par Maruyama (1990). La volonté de tirer partie des algorithmes basés sur les graphes pour les adapter à l'analyse en dépendances est à l'origine de cette méthode.

Le principe de l'analyse par satisfaction de contraintes est d'éliminer les arcs non-conformes à des contraintes pré-établies et/ou à conserver les arcs les plus probables d'un graphe complet sur une phrase. Cela revient souvent à trouver un arbre couvrant maximum. Les travaux étudiant cette méthode font alors l'hypothèse qu'un arbre couvrant maximal sur une phrase de départ est un arbre de dépendances pour la phrase. On se borne ici à considérer des arbres couvrants sur des graphes connexes.

Définition 8 *Arbre couvrant*
Un arbre couvrant, pour un graphe $G = (V,A)$ où V est un ensemble de nœuds et A un ensemble d'arcs, est un graphe $G' = (V,A')$ tel que :

- $A' \subset A$;
- *G' est connexe et ne contient pas de cycles.*

Définition 9 *Arbre couvrant maximum*
Un arbre couvrant maximum, pour un graphe pondéré, est un arbre couvrant tel que la somme des poids de ses arcs soit maximale.

Le problème de départ de l'analyse par satisfaction de contraintes est d'estimer les poids des arcs. Les poids des arcs sont en pratique évalués à l'aide d'un algorithme d'apprentissage entraîné sur les données d'un corpus en dépendances. Le but est alors de trouver un arbre couvrant maximum i.e. ayant un nombre minimal d'arcs et un poids maximal. Un algorithme connu pour la

[1] Il est à noter qu'il existe également des méthodes d'apprentissage automatique des grammaires.

résolution de ce problème est l'algorithme Chu-Liu-Edmonds (Chu et Liu, 1965; Edmonds, 1967). Cet algorithme permet en outre d'obtenir des graphes de dépendances non-projectifs.

L'algorithme 2 expose **l'algorithme de Chu-Lui-Edmonds** extrait de Kübler *et al.* (2009). L'algorithme prend en paramètre le graphe complet d'une phrase donnée et une fonction renvoyant les poids des arcs préalablement calculés. Ensuite l'algorithme parcourt récursivement les cycles du graphe pour éliminer les arcs les moins probables tout en conservant la connexité du graphe (en considérant l'orientation des arcs). La complexité de cet algorithme est en $O(n^3)$.

Algorithme 2 : CHU-LUI-EDMONDS

Données : $G = (V,A)$ un graphe, λ une fonction renvoyant le poids maximum entre deux noeuds : $\lambda_{(w_i,w_j)}$ =max $\lambda_{(w_i,d,w_j)}$ avec w_i,$w_j \in V$ et $d \in A$

$A' = \{(w_i,w_j) \mid w_j \in V, w_i = \text{argmax}_{w_i}\ \lambda_{(w_i,w_j)}\}$
$G' = (V,A')$
si G' *ne contient pas de cycles* **alors** retourner G'
Soit A_c un cycle $\in G'$
$< G_c,w_c,ep >=$ CONTRACT (G',A_c,λ)
$G =$ CHU-LUI-EDMONDS(G_c, λ)
pour l'arc $(w_i,w_c) \in A$ où $ep(w_i,w_c) = w_j$, identifier l'arc $(w_k,w_j) \in C$
trouver tous les arcs $(w_c,w_l) \in A$
$A = A \cup \{(ep(w_c,w_l),w_l)\}_{pour\ tout(w_c,w_l)\in A} \cup A_c \cup \{(w_i,w_j)\} - \{(w_k,w_j)\}$
$V = V$

Algorithme 3 : CONTRACT

Données : $G = (V,A)$ un graphe, $C \subset G$ un cycle , λ une fonction

Soit $G_c = G - C$
Ajouter un noeud w_c à G représentant le cycle C
pour tous les $w_j \in V - V_C$ **faire**
  si $\exists\ w_i \in C$ **tel que** $(w_i,w_j) \in A$ **alors**
    Ajouter un arc (w_c,w_j) à G_c **avec**
      $ep(w_c,w_j) = \text{argmax}_{w_i \in C}\ \lambda(w_i,w_j)$
      $w_i = ep(w_c,w_j)$
      $\lambda(w_c,w_j) = \lambda(w_i,w_j)$
pour tous les $w_i \in V - V_C$ **faire**
  si $\exists\ w_j \in C$ **tel que** $(w_i,w_j) \in A$ **alors**
    Ajouter un arc (w_i,w_c) à G_c **avec**
      $ep(w_i,w_c) = \text{argmax}_{w_j \in C}\ [\lambda(w_i,w_j) - \lambda(a(w_j),w_j)]$
      $w_j = ep(w_i,w_c)$
      $\lambda(w_i,w_c) = [\lambda(w_i,w_j) - \lambda(a(w_j),w_j) + \text{SCORE}(C)]$
    où $a(w)$ est le prédécesseur de w dans C et $\text{SCORE}(C) = \Sigma_{w\in C}\ \lambda(a(w),w)$
retourner $< G_c,w_c,ep >$

Parmi les travaux exploitant la méthode d'analyse par satisfaction de contraintes, on peut citer ceux de Harper et Helzerman (1995), Schröder *et al.* (2001) ou Mcdonald *et al.* (2005). L'analyseur *MST Parser* (McDonald *et al.*, 2005), débuté par Baldrige et McDonald, ou le *Mate Parser* de Bohnet (2010) utilisent également cette méthode basée sur les graphes pour l'analyse en dépendances.

Par ailleurs, notons que les premiers travaux d'analyse par satisfaction de contraintes exploitaient uniquement des traits de premier ordre dans le calcul de l'arbre couvrant maximum, c'est à dire que le poids de l'arbre correspond à une somme calculée à partir des scores des dépendances

eux-mêmes calculés indépendamment les uns des autres. Hors, les dépendances syntaxiques d'un arbre ont couramment des affinités particulières les unes avec les autres. Il est donc intéressant de prendre en compte, lors de l'apprentissage et lors de l'analyse, les poids pour des traits d'ordre plus élevé. Par exemple, McDonald et Pereira (2006) utilisent des traits de second ordre, c'est à dire qu'ils considèrent les scores de probabilité pour les dépendances adjacentes (i.e. des dépendances ayant le même gouverneur et dont les subordonnés sont consécutifs dans la phrase). Zhang et McDonald (2012) investiguent également une méthode permettant de prendre en compte des traits d'ordre supérieur.

• L'analyse par transition

L'analyse par transition consiste, à partir d'une phrase donnée à analyser, à trouver et à appliquer une bonne séquence de transitions dans le but de construire une structure de dépendances correcte pour cette phrase. Le principe est de parcourir la phrase pour trouver les dépendances à assigner entre les mots. Ces dépendances sont assignées par l'usage des transitions. Les transitions permettent également d'effectuer d'autres opérations lors de l'analyse. Elles sont propres au système par transition employé pour l'analyse.

Définition 10 *Système par transition*
Un système par transition est un quadruplet $\langle T, C, c_0, C_t \rangle$ *où :*

- T *est un ensemble de transitions ;*
- C *est un ensemble de configurations ;*
- $c_0 \in C$ *est la configuration initiale ;*
- $C_t \subset C$ *est l'ensemble des configurations finales.*

L'analyse d'une phrase donnée débute par l'affectation de cette phrase à la configuration initiale du système. Les configurations figurent l'état de l'analyse (i.e. l'état du système par transition) à un moment donné. Généralement une configuration comprend, au minimum, des mots de la phrase à analyser (indiquant la position dans le parcours de la phrase) et une structure de dépendances en cours de construction. Les configurations peuvent être modifiées par l'usage des transitions. Les transitions sont des opérations (admettant des paramètres) qui permettent de passer d'une configuration à une autre, par exemple en ajoutant des arcs à la structure de dépendances ou en modifiant la position des mots. À partir de la configuration initiale du système, le but d'une analyse par transition est de prédire et d'appliquer les transitions qui permettront de passer d'une configuration à une autre jusqu'à obtenir une configuration finale dans laquelle est contenue la structure de dépendances correcte pour la phrase à analyser. En ce sens, les configurations pouvant être vues comme les états d'un automate, l'analyse par transition peut alors également être définie comme le problème de recherche du chemin optimal dans un automate fini (Nivre, 2008). De ce point de vue, les transitions permettent de passer d'un état à un autre dans l'automate.

Les systèmes par transitions sont divers. Il se différencient les uns des autres par la variabilité des transitions et des configurations qu'ils emploient. Techniquement, une configuration est une structure de mémoire abstraite pouvant utiliser les piles, les compteurs, etc. Si bien que Nivre (2008) propose deux classifications principales pour les systèmes par transition : *stack-based* (basés sur les piles) et *list-based* (basés sur les listes). Il en résulte des algorithmes d'analyse de complexité plus ou moins grande et permettant d'engendrer des structures projectives ou non-projectives. Dans cette section, nous nous restreignons dans un premier temps à définir les éléments des systèmes par transitions tels qu'utilisés lors des prémices de l'analyse en dépendances (projective) par transition proposés dans la littérature.

Définition 11 *Configuration*
On définit une configuration par un triplet $\langle \sigma, \beta, A \rangle$ *où :*

- σ *est une pile de mots (déjà traités ou partiellement traités) ;*
- β *est une mémoire tampon contenant les mots en attente de traitement ;*
- A *est un ensemble d'arcs correspondant aux dépendances de la structure finale.*

Les premières définitions des configurations, telles que définies ci-dessus, incluaient une pile et une mémoire tampon pour sauvegarder les mots et une structure de donnée permettant de mémoriser les dépendances. Les mots sont, au début de l'analyse, empilés dans la mémoire tampon. Une analyse par transition, pour une phrase $W = w_1...w_n$, débute par une configuration de la forme $\langle [0], [1,...,n], \emptyset \rangle$ où 0 est l'indice de la racine[2] de la structure de dépendances. Les mots de la mémoire tampon peuvent être dépilés pour être empilés successivement sur la pile ou supprimés selon les transitions appliquées, jusqu'à épuisement des mots dans la mémoire tampon. La configuration finale de l'analyseur doit être de la forme $\langle [0], [], A \rangle$. L'analyse est alors l'application d'une suite de transitions de la configuration initiale à la configuration finale.

Parmi les premiers travaux d'analyse par transition appliqués à l'analyse en dépendances on peut trouver ceux de Kudo et Matsumoto (2002) et Yamada et Matsumoto (2003). Ils se sont tout d'abord intéressés, pour le chinois et l'anglais en particulier, à l'attachement des dépendances sans en chercher les étiquettes. Le formalisme de base de l'analyse par transition propose trois transitions principales :

- ***LEFT-ARC*** (arc gauche) ajoute une dépendance gauche entre le mot du haut de la pile σ et le premier mot de la mémoire tampon β et supprime le mot du haut de la pile (le subordonné dans la dépendance) ;
- ***RIGHT-ARC*** (arc droit) ajoute une dépendance droite entre le mot du haut de la pile σ et le premier mot de la mémoire tampon β, supprime le premier mot de la mémoire tampon (le subordonné dans la dépendance) et ajoute le mot du haut de la pile à la place ;
- ***SHIFT*** (décalage) dépile un mot de la mémoire tampon β pour l'empiler sur la pile σ.

Ces transitions, leurs conditions d'application et leurs effets sur les configurations sont exposés formellement dans la table 2.1.

Transition	**Effets sur les configurations**	**Conditions**
Left-Arc(l)	$(\sigma \mid i,j \mid \beta, A) \Rightarrow (\sigma,j \mid \beta, A \cup \{(j,l,i)\})$	$i \neq 0 \wedge \neg\exists k \; \exists l' \; (k,l',i) \in A$
Right-Arc(l)	$(\sigma \mid i,j \mid \beta, A) \Rightarrow (\sigma,i \mid \beta, A \cup \{(i,l,j)\})$	$\neg\exists k \; \exists l' \; (k,l',j) \in A$
Shift	$(\sigma,i \mid \beta, A) \Rightarrow (\sigma \mid i,\beta,A)$	

TABLE 2.1 – Transitions de la méthode de base de l'analyse par transition.

L'analyse par transition est couvrante et non-déterministe si l'on considère le formalisme de base tel quel. Néanmoins, les systèmes par transition sont couramment couplés à des outils de prédiction des transitions. Généralement, le but est de prédire, pour chaque nouvelle configuration crée, une unique transition à appliquer. Ainsi, on obtient des systèmes permettant de construire rapidement une structure de dépendances pour chaque phrase donnée à analyser. L'analyse par

[2]La racine de la structure de dépendances est une racine « artificielle » qui permet d'ancrer le mot racine de la phrase i.e. de lui assigner un pseudo-gouverneur.

transition profite alors des méthodes d'apprentissage automatique pour entraîner les systèmes sur les données de corpus en dépendances. Les configurations sont traduites en vecteurs de traits et les transitions sont prédites en fonction de leurs probabilités vis-à-vis des configurations à un instant donné. L'analyse devient déterministe modulo un oracle stochastique et s'effectue en temps linéaire.

Différentes applications et améliorations, dont la possibilité de gérer les dépendances non-projectives (Attardi, 2006), ont été présentées au cours des années. On peut citer les travaux de Nivre (2003, 2006) ou des travaux plus récents de Huang et Sagae (2010) ou Choi et Palmer (2011). Parmi les méthodes les plus connues et utilisées dans le domaine de l'analyse en dépendances par transitions, citons en particulier celles ayant été développées pour le MaltParser (Nivre *et al.*, 2006a). Par exemple, pour l'analyse en dépendances projective, il est possible d'employer l'une des méthodes suivantes, chacune de complexité linéaire :

- *arc-standard*, correspondant à la méthode de base présentée précédemment ;
- *arc-eager*, une méthode étendue qui attache les dépendances droites dès que possible et inclut la transition **REDUCE** (permettant de supprimer le mot du haut de la pile si celui-ci a déjà un gouverneur) ;
- *stackproj*, une version proche de *arc-standard* ;
- *planar*, une méthode permettant de produire des graphes en dépendance planaires.

Pour l'analyse en dépendances non-projective, le MaltParser propose également les méthodes suivantes :

- *covnonproj*, une méthode (de complexité quadratique) inspirée par Covington (2001) ;
- *stackeager* et *stacklazy*, des méthodes dites *stack-based* utilisant une transition appelée **SWAP** (permettant d'échanger l'ordre des deux mots du haut la pile). L'ajout de cette transition rend leur complexité algorithmique quadratique bien qu'en pratique les tests témoignent d'une exécution en temps linéaire ;
- *2planar*, une méthode (de complexité linéaire) permettant d'analyser des graphes bi-planaires (i.e. dont les nœuds peuvent être colorés avec seulement 2 couleurs sans que deux nœuds adjacents aient la même couleur). Une grande partie des phénomènes syntaxiques non-projectifs peuvent être représentés par des graphes bi-planaires mais pas la totalité.

De plus, le MaltParser inclut un mode pseudo-projectif (Nivre et Nilsson, 2005) permettant d'utiliser les méthodes projectives sur des données non-projectives. Le principe de la méthode est de convertir les structures de dépendances non-projectives en structure de dépendances projectives, d'employer des méthodes d'analyse projectives pour prédire les dépendances, puis de retrouver les dépendances non-projectives automatiquement grâce aux informations encodées dans les étiquettes des dépendances. L'algorithme 4 présente la méthode de projectivisation d'une structure de dépendances non-projective. L'algorithme emploie deux fonctions :

- SMALLEST-NONP-ARC(A) qui retourne l'arc ayant la plus courte distance avec son gouverneur parmi les arcs non-projectifs de l'ensemble A ;
- LIFT$(a =(w_j \rightarrow w_k))$ qui retourne la dépendance $w_i \rightarrow w_k$ si la dépendance $w_i \rightarrow w_j$ existe i.e. le nouveau gouverneur de w_k est un gouverneur de son ancien gouverneur.

Les arcs projectifs substitués sont couplés à des étiquettes en dépendances contenant une information encodée (selon différents modes d'encodage : *baseline*, *head*, *path*) qui permet de retrouver avec plus au moins de précision la dépendance non-projective à laquelle elle est associée. La méthode pseudo-projective dans son ensemble, i.e. la projectivisation combinée à une méthode d'analyse projective, permet d'obtenir des résultats équivalents voir meilleurs que ceux des méthodes non-projectives.

Algorithme 4 : PROJECTIVISATION

Données : $G = (V,A)$ un graphe de dépendances
$A' \leftarrow A$
tant que *G est non-projectif* **faire**
  $a \leftarrow$SMALLEST-NONP-ARC(A')
  $A' \leftarrow (A' - \{a\}) \cup \{\text{LIFT}(a)\}$
retourner $G = (V,A')$

2.4 Les corpus en dépendances

Pour évaluer/comparer les différentes méthodes d'analyse en dépendances proposées à travers de multiples travaux, il est nécessaire d'expérimenter ces méthodes sur des données validées. Ces données sont rassemblées sous forme de corpus (i.e. ensemble de phrases). Lorsque chaque phrase d'un corpus est associée à une structure syntaxique on appelle ces ensembles des banques d'arbres ou corpus arborés ou *Treebanks*. Si les arbres se trouvent être des structures de dépendances, le terme « corpus en dépendances » peut-être employé. Dans cette section, nous faisons donc état des corpus en dépendances existants et disponibles à des fins de recherche, pour le français ainsi que pour différentes langues et détaillons les différentes étapes des processus d'annotation ayant permis de construire ces corpus.

2.4.1 Corpus pour le français

Le corpus probablement le plus connu dans le domaine de la syntaxe est le French Treebank (FTB) (Abeillé *et al.*, 2003) ou Paris 7 Treebank (P7T). Ce corpus est composé de 24000 phrases et fut annoté initialement en constituants puis converti automatiquement en dépendances (Candito *et al.*, 2009). Le corpus est disponible sous licence. Les phrases sont extraites du journal LeMonde (1989-1993). Malgré les thèmes différents abordés (économie, littérature, politique, etc.) par le journal, le style grammatical reste restreint au domaine journalistique. Une méthode d'analyse syntaxique dirigée par les données dont le modèle est entraîné sur ces données atteindra alors très sûrement de meilleurs scores sur des données de même nature que sur de nouvelles données (hors-domaine).

Ce problème constitue l'argument principal à l'origine de la construction du corpus Sequoia (Candito et Seddah, 2012a). Il s'agit d'un corpus annoté suivant un schéma d'annotation proche de celui du FTB, donc en constituants. Dans ce corpus les phrases proviennent de 4 sources différentes :

- Europarl ;
- l'agence européenne du médicament ;
- l'Est Républicain ;

- Wikipédia Fr.

Il en résulte un corpus annoté de 3204 arbres. Un exemple d'arbre syntagmatique pour une phrase du corpus Europarl de Sequoia est donné par la figure 2.9.

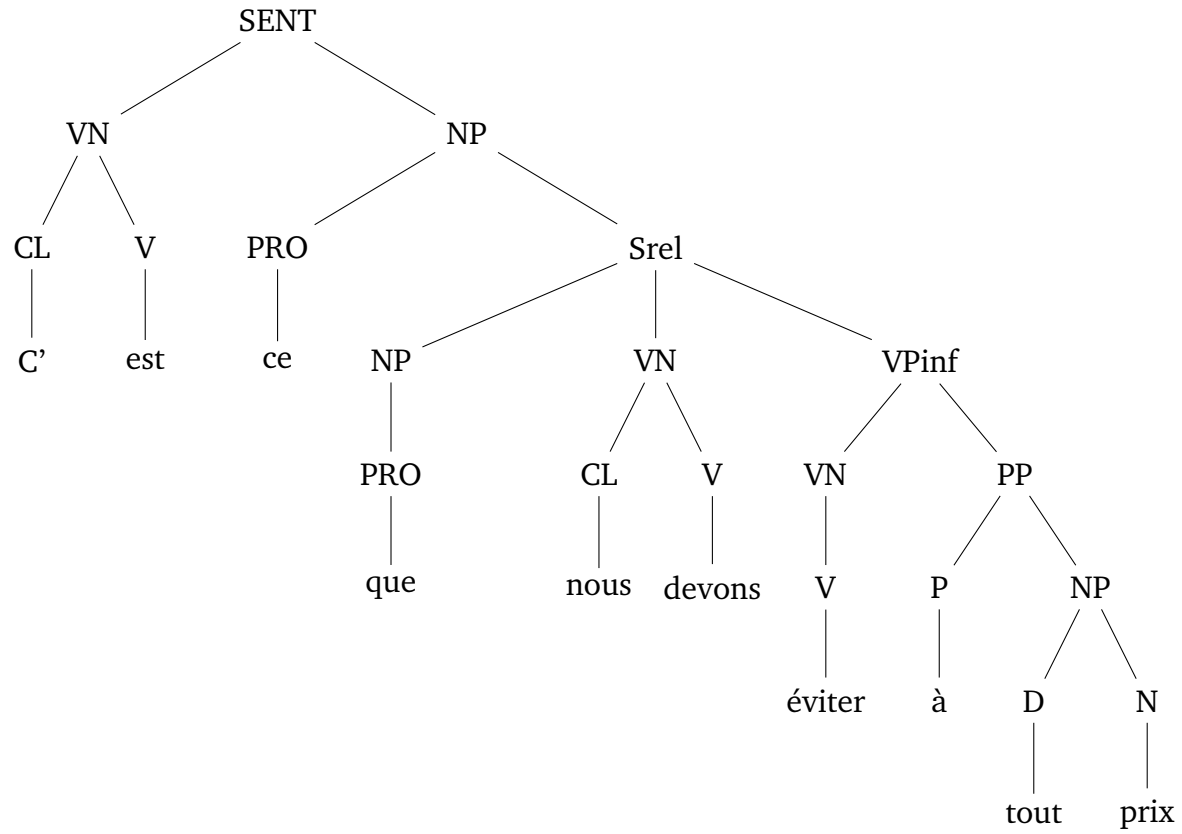


FIGURE 2.9 – Arbre syntagmatique pour la phrase « C'est ce que nous devons éviter à tout prix » extraite du corpus Sequoia (Europarl).

Les processus d'annotations du FTB et de Séquoia suivent un déroulement similaire dans le sens où ils incluent alternativement des étapes de pré-annotation automatique et des étapes de validation manuelle, mais ils diffèrent sur certains points qui ont été améliorés pour l'annotation de Sequoia. Nous décrivons dans un premier temps les étapes effectuées lors de la construction du FTB :

- segmentation : segmentation en phrase et tokenisation (le tokeniseur utilise un lexique pour reconnaître les mots composés (ou expressions multi-mots) et conserve la plus longue séquence possible correspondant à une unité de sens ;
- étiquetage des parties du discours (POS pour *Part-Of-Speech*) : lors de la 1ère phase, chaque mot reçoit l'étiquette la plus probable (sa probabilité est estimée à l'aide d'une méthode basée sur les trigrammes). Lors de la deuxième phase, des règles contextuelles sont construites à la main pour modifier certaines étiquettes ;
- validation de la segmentation et de l'étiquetage des POS par les annotateurs ;
- ajout automatique des lemmes ;

- regroupement d'items formant des dates, nombres ou titres à l'aide d'expressions régulières et vérification (i.e. ajout manuel des groupes oubliés) ;
- annotation syntaxique automatique permettant d'indiquer les frontières des constituants et d'annoter les fonctions syntaxiques des mots (i.e. annotation fonctionnelle des dépendants de verbes) et des constituants.
- validation manuelle des constituants.

Pour la construction du corpus Sequoia, les auteurs ont tout d'abord choisi d'utiliser des outils différents et un jeu d'étiquettes étendu. En ce qui concerne le schéma d'annotation, les différences se font sur :

- la segmentation des mots composés, la séparation en mot simple est conservée dans les phrases du Séquoia si l'expression est syntaxiquement régulière ;
- l'étape d'étiquetage des POS, comprenant simplement une étape de pré-annotation automatique à l'aide d'un étiqueteur et une étape de validation manuelle ;
- l'annotation syntaxique, une première étape automatique d'analyse syntaxique (en constituants) est effectuée par deux analyseurs différents en tenant compte des POS précédemment assignés et validés puis une seconde étape de validation manuelle est effectuée indépendamment par deux annotateurs pour les deux sorties, enfin une étape d'adjudication des annotations est réalisée ;
- l'annotation des fonctions des dépendants des verbes, incluant une étape automatique pour assigner ces fonctions, une étape de correction par les deux annotateurs et l'adjudication des annotations.

Ces deux corpus ne sont pas seulement cohérents vis-à-vis du schéma d'annotation mais ont aussi été convertis automatiquement en dépendances[3] (Candito *et al.*, 2009) suivant les mêmes étapes.

La conversion en dépendances comprend les étapes suivantes :

- la décomposition des mots composés syntaxiquement réguliers ;
- l'extension du jeu d'étiquettes (pour le FTB) ;
- le pré-traitement des prépositions et compléments dans l'arbre en constituants pour les élever au rang de tête de constituants, assurant par ailleurs la projectivité de la future structure en dépendances ;
- l'application d'une méthode récursive de propagation des têtes (Magerman, 1995) permettant d'obtenir les arbres en dépendances.

[3] Notons qu'il s'agit d'une conversion en dépendances de surface (voir 2.2.1) mais que le corpus Sequoia a également été converti en dépendances profondes (Candito *et al.*, 2014).

Cette conversion automatique assure la projectivité des arbres en dépendances bien que certaines constructions grammaticales nécessitent l'intégration de dépendances non-projectives[4]. Le corpus FTB converti en dépendances est donc totalement projectif tandis que le corpus Sequoia contient des cas non-projectifs qui furent corrigés manuellement après conversion (Candito et Seddah, 2012b). La figure 2.10 présente la structure de dépendances non-projective pour la phrase donnée en exemple dans la figure précédente (2.9).

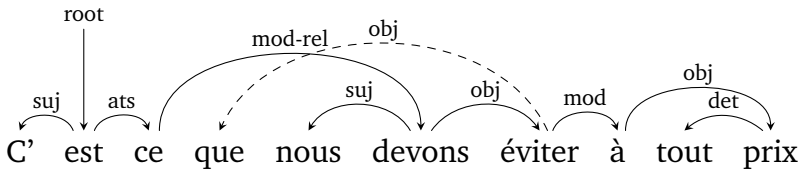


FIGURE 2.10 – Structure de dépendances non-projective pour la phrase « C'est ce que nous devons éviter à tout prix » extraite du corpus Sequoia (Europarl).

Pour trouver plus de dépendances non-projectives dans les corpus librement disponibles pour le français, il faut se tourner vers le projet *Universal Dependency Treebank* (UDT) (McDonald *et al.*, 2013). Il s'agit d'un ensemble de corpus pour diverses langues annotées suivant le schéma d'annotation en dépendances standard de Stanford (de Marneffe et Manning, 2008) proposé en premier lieu pour l'anglais. La représentation en dépendances de Stanford autorise à l'origine la double gouvernance, i.e. un mot peut avoir deux gouverneurs différents. Par conséquent, la structure peut contenir des cycles. Cependant, une représentation (acyclique) dans laquelle chaque mot a un et un seul gouverneur fut ensuite proposée dans le but d'obtenir une représentation adaptée aux analyseurs classiques du domaine ; il s'agit du schéma d'annotation standard. Par ailleurs, une représentation appelée *content-head* est aussi proposée pour certaines langues dont le français. Dans cette représentation, les mots porteurs de sens sont considérés comme des têtes syntaxiques, e.g. l'objet d'un verbe copule sera considéré comme étant la racine de la phrase et le verbe dépendra de cet objet. En particulier, les objets des verbes copules et les appositions sont des têtes syntaxiques. Toutefois, on s'intéressera ici particulièrement à la représentation en dépendances standard.

L'objectif du projet UDT était de rendre disponible des corpus en dépendances dans différentes langues dont le jeu d'étiquettes (i.e. les étiquettes POS et les étiquettes des dépendances) est identique. Chaque corpus fut donc annoté suivant le même schéma d'annotation pour les étiquettes POS, le *Google Universal Part-Of-Speech Tags* (Petrov *et al.*, 2012), et pour les dépendances, une variante du jeu d'étiquettes utilisé pour l'anglais dans la représentation en dépendances standard. Le processus d'annotation global fut différent selon les langues. Certains ensembles de données, tels que les corpus suédois et anglais, furent convertis des constituants aux dépendances. Les autres corpus, dont le corpus français, furent manuellement annotés. En premier lieu, les annotateurs proposèrent des jeux d'étiquettes en dépendances adaptés à chaque langue mais restant très proche du jeu d'étiquettes pour l'anglais. Puis, les jeux d'étiquettes furent harmonisés pour conserver le moins d'étiquettes possibles[5]. Puis, l'étape de segmentation en tokens se sert d'un système basé sur des règles. Les annotateurs sont ensuite chargés de sélectionner les phrases pour mettre de côté celles qui seraient incomplètes (dues aux erreurs de segmentation), incompréhensibles ou trop courtes. Puis les phrases sont automatiquement étiquetées (Das et Petrov, 2011), parsées (McDonald *et al.*, 2011) et manuellement validées. Parmi les lignes directives du schéma d'annotation,

[4]Lors de la conversion du FTB en dépendances (Candito *et al.*, 2009), 120 phrases furent annotées manuellement en dépendances dans le but d'évaluer le taux d'erreurs engendrés par la conversion automatique. Sur cet ensemble de données, 1,22% des dépendances ont été annotées comme dépendances non-projectives.

[5]Une étiquette spécifique à une langue est préservée seulement si le phénomène syntaxique auquel elle fait référence ne peut être renseigné par aucune autre étiquette pré-existante.

aucune restriction n'est faite sur les phénomènes discontinus des langues naturelles. Le corpus français contient donc environ 12% de phrases pour lesquelles les structures en dépendances associées sont non-projectives.

Nous rassemblons dans le tableau 2.2 quelques statistiques et propriétés des différents corpus pour le français dont nous avons discuté dans cette sous-section. Notons également que nous nous sommes restreint à présenter des corpus en dépendances pour le français écrit mais qu'il existe également des corpus en dépendances pour le français parlé tels que le corpus Rhapsodie (Lacheret *et al.*, 2014) ou le corpus développé par Cerisara *et al.* (2010) à partir du corpus ESTER contenant des transcriptions manuelles d'émissions de radio.

Corpus	Nb phrases	Nb tokens	Non-Projectif ?
FTB dép.	24 000	780 000	non
Sequoia dép.	3 204	69 246	oui
UDT French	16 422	396 511	oui

TABLE 2.2 – Statistiques de différents corpus en dépendances disponibles pour le français écrit.

2.4.2 Corpus pour diverses langues

Il existe, pour diverses langues, des corpus en dépendances disponibles, projectifs et non-projectifs selon les besoins syntaxiques de la langue, incluant un nombre important de données. Le projet UDT (*Universal Dependency Treebank*) (McDonald *et al.*, 2013), incluant le corpus français dont nous avons parlé dans la section précédente, met également à disposition des corpus pour le portugais (brésilien), l'anglais, le finnois, l'allemand, l'italien, l'indonésien, le japonais, le coréen, l'espagnol et le suédois. Les annotations de ces corpus sont harmonisées dans le but d'effectuer plus aisément des travaux multilingues.

Cependant, la plupart des corpus en dépendances de grande envergure suivent leur propre schéma d'annotation, spécifique à une langue ou à une théorie. Par exemple, le *Prague Dependency Treebank* (PDT) (Hajičová *et al.*, 2001) comprend 3 niveaux d'annotations : morphosyntaxique, analytique (syntaxe) et tectogrammatical (sémantique). Il s'agit du corpus en dépendances le plus important, en terme de taille, comprenant plus de 90 000 phrases en langue tchèque. La langue tchèque étant relativement flexible, le choix d'une annotation en dépendances s'avère bien adapté. Par ailleurs, un corpus en dépendances pour la langue arabe a été développé suivant le même schéma d'annotation que le PDT, il s'agit du *Prague Arabic Dependency Treebank* (Hajič et Zemánek, 2004).

La syntaxe de l'allemand s'interprète de même assez naturellement grâce à la représentation en dépendances. Le *Tiger Treebank* (Brants *et al.*, 2002) est un corpus en dépendances pour l'allemand comprenant environ 50 000 phrases annotées.

Parmi les langues dont l'ordre des mots est flexible on peut aussi trouver le russe, langue pour laquelle le corpus *SynTagRus* (Boguslavsky *et al.*, 2000, 2009), comprenant plus de 40 000 phrases annotées en dépendances, a été développé. Une grande part des étiquettes en dépendances utilisées dans ce corpus proviennent de la théorie Sens-Texte de Mel'cuk (1988).

Généralement, ces corpus ont été construits semi-automatiquement. Certains sont tout d'abord passés par une phase d'annotation totalement manuelle. Cependant, dès lors que la taille des données devient suffisamment importante, il est intéressant d'introduire des outils automatiques permettant d'alléger le travail des annotateurs. Avec la construction de corpus est alors souvent

combinée le développement d'outils d'étiquetage, d'analyse, etc. Ces outils permettent d'effectuer plus rapidement certaines étapes mais une validation manuelle intermédiaire ou finale est souvent requise pour certifier l'exactitude des annotations.

En outre, d'autres corpus en dépendances disponibles n'ont pas été manuellement annotés mais automatiquement convertis. Dans ce cas, il n'y a pas toujours d'étape de validation manuelle des structures produites. La conversion consiste à convertir les arbres d'un corpus depuis des structures en constituants vers des structures en dépendances. Il s'agit habituellement de corpus dans des langues dont l'ordre des mots n'est pas flexible. La raison vient du fait que pour ces langues, une représentation syntaxique en constituants paraissait plus adaptée à l'origine et résultait de travaux sur les grammaires dans la lignée de Chomsky. Les corpus français *French Treebank* et *Séquoia* dont nous avons discuté dans la section précédente ont été annotés en constituants. Les conversions de ces corpus ont été inspirées par une méthode de conversion (de Marneffe *et al.*, 2006) appliquée tout d'abord à un corpus anglais, le *Penn Treebank* (Marcus *et al.*, 1993). Pour l'anglais il existe par ailleurs le *PARC 700 Dependency Bank* (King *et al.*, 2003) comprenant, comme son nom l'indique, 700 phrases annotées semi-automatiquement en dépendances.

2.4.3 Campagnes d'évaluation

La diversité des schémas d'annotation et formats utilisés par les corpus en dépendances peut rendre difficile les comparaisons des analyseurs ou méthodes d'analyses en dépendances entre différentes langues. Les campagnes d'évaluations (*Shared Tasks*) permettent alors d'homogénéiser les données dans le but d'évaluer équitablement les performances des analyseurs. Globalement, il s'agit de proposer une tâche particulière à résoudre, en ce qui nous concerne, dans le domaine de l'analyse en dépendances. Les organisateurs de la campagne fournissent[6] les données nécessaires, les conditions de la tâche et généralement les programmes permettant d'évaluer les résultats. Les données, généralement multilingues dans le domaine de l'analyse en dépendances, sont distribuées sous un format standard. Il s'agit couramment de données récupérées à partir de corpus existants qui ont été converties, retravaillées, reformatées. Ces données restent souvent disponibles après la campagne d'évaluation, librement ou par l'intermédiaire de « catalogues de données ». Nous présentons ici quelques exemples de campagnes d'évaluation ayant fourni des données pour l'analyse en dépendances.

Les *CoNLL Shared Tasks* sont particulièrement connues pour proposer régulièrement des tâches différentes dans le domaine du traitement automatique de la langue. En ce qui concerne l'analyse en dépendances, la *CoNLL-X Shared Task* (Buchholz et Marsi, 2006) fut la première à proposer une tâche d'analyse en dépendances multilingue. Les données des corpus disponibles pour cette tâche proviennent de différents corpus en dépendances existants (pour le tchèque, l'arabe, le slovène, le danois, le suédois et le turc) ou de corpus en constituants convertis en dépendances (pour l'allemand, le japonais, le portugais, le hollandais, le chinois, l'espagnol et le bulgare). Tous ces corpus (excepté le corpus pour le chinois) contiennent des dépendances non-projectives.

Parmi les différentes *CoNLL Shared Tasks* proposées au cours des années il y a eu également la *CoNLL Shared Task 2009* (Hajič *et al.*, 2009) présentant une tâche dans le domaine de l'analyse en dépendances. Les corpus fournis lors de cette campagne incluent des données pour les langues catalane, espagnole, chinoise, tchèque, anglaise, allemande et japonaise.

De fait, le format CoNLL, dans lequel les données sont fournies lors des *CoNLL Shared Tasks*, est aujourd'hui largement utilisé comme format standard pour les entrées des analyseurs en dépendances. La *SPMRL Shared Task 2013* (Seddah *et al.*, 2013b) a également mis à disposition des

[6]Les données ne sont pas toujours libres mais nécessitent parfois d'obtenir une licence particulière pour pouvoir les utiliser et partager les résultats obtenus par l'évaluation de ces données.

données dans ce format lors de sa campagne. Il s'agissait alors de corpus en dépendances pour l'arabe, le basque, le français, l'allemand, l'hébreu, le hongrois, le coréen, le polonais et le suédois.

2.5 Conclusion

La représentation de la syntaxe des langues en dépendances découle de différentes théories syntaxiques privilégiant la description de liens entre les mots d'une phrase plutôt que leur regroupement en syntagme. Néanmoins il n'existe pas une unique manière de représenter les dépendances. Une représentation standard, fréquemment employée et communément désignée par le terme d'arbre de dépendances admet des propriétés de base généralement partagées dans le domaine de l'analyse en dépendances. Elle permet en particulier de prendre en compte la notion de non-projectivité qui nous intéresse notablement dans nos travaux car elle permet de décrire l'ensemble des phénomènes syntaxiques des langues.

Avec l'avènement de l'informatique sont arrivées des méthodes d'analyse en dépendances permettant de produire automatiquement de telles représentations en dépendances pour des phrases du langage naturel. Les méthodes basées sur les grammaires furent beaucoup étudiées dans le domaine de l'analyse syntaxique en constituants mais également pour l'analyse en dépendances. Les méthodes dites dirigées par les données sont plus récentes mais jouissent d'une popularité grandissante dans le domaine du traitement automatique de la langue. Ces méthodes ont chacune leurs inconvénients et leurs avantages et continuent d'être largement étudiées. Nous nous intéressons à chacune de ces méthodes dans le cadre de nos travaux, d'une part pour le développement de corpus en dépendances non-projectifs et d'autres part pour l'amélioration de la prédiction des dépendances non-projectives dans le cadre de l'analyse en dépendances.

En outre, nous avons vu qu'il existe des corpus en dépendances disponibles pour le français mais que ceux-ci ne prennent pas ou peu en compte les phénomènes discontinus de la langue française par des dépendances non-projectives. Néanmoins, il existe des corpus en dépendances pour d'autres langues comportant des dépendances non-projectives en nombre suffisant pour s'attarder sur le cas de ces dépendances à travers les méthodes d'analyse en dépendances.

3

Les méthodes dirigées par les données dans les processus d'analyse syntaxique

3.1 Introduction

L'analyse syntaxique, en tant que problème informatique à résoudre, consiste à annoter automatiquement et syntaxiquement les phrases des langues naturelles. De multiples travaux dans le domaine ont proposé des méthodes permettant de résoudre ce problème. Il est courant de voir ces méthodes catégorisées soit dans le champ des méthodes basées sur des règles soit dans le champ des méthodes dirigées par les données (bien que la combinaison des deux soit également possible). Les méthodes basées sur des règles utilisent des règles syntaxiques manuellement ou automatiquement créées, décrivant la syntaxe des langues, pour diriger les analyses. Les **méthodes dirigées par les données** sont des méthodes qui exploitent les informations provenant d'ensemble de données annotés (grammaticalement, syntaxiquement, etc) pour annoter de nouvelles données. En particulier, ces méthodes exploitent les outils provenant de l'apprentissage automatique supervisé pour effectuer certaines étapes des analyses syntaxiques. Alors, avec la production et la disponibilité grandissante de larges corpus annotés dans de multiples langues il est de plus en plus commode d'utiliser des méthodes dirigées par les données dans le domaine de l'analyse en dépendances car plus les données à exploiter sont importantes (en terme de taille) et variées (i.e. venant de sources différentes), plus les méthodes dirigées par les données permettent d'atteindre des performances intéressantes en terme de précision.

L'apprentissage automatique fait partie intégrante des méthodes dirigées par les données. Les outils d'apprentissage automatique sont généralement utilisés dans le contexte de l'apprentissage et de la prédiction d'étiquettes. Le terme d'**étiquette** englobe différents éléments. Dans le domaine du traitement automatique des langues, les étiquettes peuvent indiquer par exemple des noms de catégories (grammaticales, syntaxiques) ou des opérations à effectuer (e.g. ajout d'un arc dans un système par transition). Une prédiction juste de ces étiquettes est donc particulièrement importante dans le traitements de multiples tâches. Dans la première section de ce chapitre, nous présentons les notions générales de l'apprentissage automatique, puis nous révélons ensuite comment l'apprentissage automatique est intégré aux différents processus nécessaires à l'analyse en dépendances et à l'analyse syntaxique de manière générale.

Notons que les évaluations des méthodes d'analyse syntaxique ne se font plus seulement sur l'étape d'analyse uniquement. En effet, les analyses réalisées à partir de données correctement pré-annotées (i.e. dont la segmentation et l'étiquetage grammatical ont été certifiés par au moins un annotateur) ne présentent pas des scores correspondant à une évaluation dans une situation réelle (i.e. analyse à partir d'une phrase brute). Ainsi, les évaluations incluent maintenant couramment les performances des pré-traitements nécessaires dans l'évaluation de l'analyse. Ces différentes étapes de pré-traitement ont elles-mêmes été beaucoup étudiées et sont impliquées dans le développement de diverses méthodes dont une part significative de méthodes dirigées par les données. Nous présentons donc, dans la seconde section de ce chapitre, les différentes tâches faisant office d'étapes de pré-traitement à l'analyse en dépendances et les différentes manières de les aborder.

Dans la troisième section, nous revenons sur les différentes méthodes d'analyse en dépendances majoritairement étudiées (voir la section 2.3 du chapitre 2) pour présenter la façon dont elles sont modifiées ou combinées avec des méthodes dirigées par les données. Dans le domaine de l'analyse en dépendances et globalement dans le domaine de l'analyse syntaxique, les méthodes dirigées par les données ont d'une part été intégrées aux méthodes existantes basées sur les grammaires et d'autre part ont donné naissance à des systèmes entièrement dirigés par les données. Nous observons donc, dans un premier temps, que les méthodes semi-dirigées par les données ont été largement étudiées à travers les grammaires probabilistes et sont également souvent associées à des étapes de pré-traitement tel que le supertagging ou de post-traitement tel que le ré-ordonnancement des analyses. Puis, les méthodes intégralement dirigées par les données telles que l'analyse par satisfaction de contraintes et l'analyse par transition sont apparues plus récemment et ont pris une part importante dans le domaine de l'analyse en dépendances.

3.2 Apprentissage automatique et modèles statistiques

L'apprentissage automatique tel qu'employé dans les méthodes dirigées par les données comprend deux étapes : une étape d'apprentissage des étiquettes et une étape de prédiction des étiquettes. Ces étiquettes peuvent être de simples catégories ou représenter des objets complexes. Leurs contenu (les informations qu'elles contiennent) est adapté aux tâches pour lesquelles elles sont utilisées. Ces étiquettes peuvent indiquer par exemple les noms des catégories grammaticales dans le cadre d'un processus d'étiquetage grammaticale ou le nom des opérations (et de leurs paramètres) à appliquer lors de l'emploi d'un système par transition[1].

Par ailleurs, les données exploitées pour l'apprentissage sont correctement annotées et les étiquettes sont connues lors de l'étape d'apprentissage[2]. En fait, les données sont des ensembles d'exemples qui sont sous la forme de paires objet/étiquette. L'objet correspond à une description des données, i.e. il désigne l'ensemble des caractéristiques d'une donnée de l'ensemble d'apprentissage. Dans le domaine du traitement automatique de la langue, les données sont généralement des mots ou des ensembles de mots. Les caractéristiques de ces données sont alors des attributs extraits à partir de ces mots, pertinents vis-à-vis du traitement de la langue, tel que le mot lui-même, ces traits morphologiques (e.g. genre, nombre, temps), son contexte. Ces caractéristiques (traits) forment un objet (une donnée) que l'on retrouvera en outre souvent sous la forme d'un vecteur de traits. Les traits à considérer sont définis manuellement ou automatiquement. Il est possible d'utiliser des programmes pour extraire automatiquement les traits pertinents vis-à-vis d'un traitement particulier.

L'étape d'apprentissage consiste à entraîner un modèle statistique à partir d'un ensemble de données. La première tâche de l'étape d'apprentissage consiste à répertorier tous les couples ob-

[1] Voir la sous-section 2.3.2 du chapitre 2 pour plus de détails sur les systèmes par transition.
[2] On se place dans le cadre d'un apprentissage automatique supervisé.

jet/étiquette pertinents de l'ensemble de données (i.e. les traits pertinents permettant de caractériser précisément les étiquettes à prédire). La mémorisation de tous les couples objet/étiquette d'un ensemble de données permet ensuite de regrouper les couples identiques ou similaires. Il s'agit d'une phase de généralisation. Le but de l'apprentissage est alors d'apprendre un modèle à partir duquel on obtiendra le meilleur taux de prédiction des étiquettes pour les objets observés. Le choix des traits, pour la description des objets, est souvent très important car il détermine la précision du modèle.

L'étape de prédiction consiste à exploiter le modèle appris pour prédire les étiquettes sur de nouvelles données. Il est possible à partir du modèle statistique entraîné d'établir la probabilité de chaque étiquette (de l'ensemble d'étiquettes possibles) en fonction des objets observés et en conséquence, d'estimer l'étiquette la plus probable pour un objet donné. Le contenu de cette étiquette peut alors être exploité ultérieurement dans le cadre d'utilisation d'une méthode dirigée par les données.

L'apprentissage automatique s'appuie généralement sur des modèles statistiques. Ces modèles sont connus et couramment utilisés dans le domaine du traitement automatique de la langue. Parmi les modèles fréquemment citées on trouve des modèles génératifs et des modèles discriminants. Par exemple, les réseaux bayésiens ou modèles graphiques probabilistes ont évolués et pris diverses formes parmi lesquelles on trouve :

- les HMM (*Hidden Markov Model* ou modèle de Markov caché) (Rabiner, 1989) ;
- les MEMM (*Maximum Entropy Markov Model*) (McCallum *et al.*, 2000) ;
- les CRF (*Conditional Random Fields* ou champs markoviens conditionnels) (Lafferty *et al.*, 2001).

Les réseaux bayésiens permettent de prédire des séquences d'étiquettes. D'autre part, les classificateurs, aussi couramment exploités, prédisent indépendamment les étiquettes. On y trouve principalement :

- le MaxEnt (*maximum entropy*) (Ratnaparkhi, 1996) ;
- le Perceptron (Rosenblatt, 1958) ;
- les SVM (*Support Vector Machine* ou séparateur à vaste marge) (Vapnik, 1998).

3.3 Pré-traitements

3.3.1 Segmentation et étiquetage grammatical

La segmentation et l'étiquetage grammatical sont des tâches qui ne sont pas toujours prises en compte lors de l'évaluation d'une procédure d'analyse syntaxique mais qui sont généralement nécessaires. Que ce soit lors de la construction de corpus ou pour l'évaluation de méthodes d'analyse, la segmentation des phrases et l'étiquetage grammatical des mots résultants sont des étapes importantes (il ne s'agit pas toujours d'étapes préliminaires car elles peuvent être combinées à l'analyse syntaxique mais en tant qu'étapes préliminaires à l'analyse elles apportent des informations pertinentes). Des méthodes automatiques sont fréquemment employées dans les processus d'annotation de corpus pour alléger le travail des annotateurs humains. En outre, ces méthodes sont aussi de plus en plus utilisées et intégrées dans les processus complets d'analyse syntaxique dans

le but d'évaluer les performances réelles des analyseurs et de fournir des mécanismes autonomes permettant d'effectuer des analyses à partir de phrases brutes.

Nous considérons ici la segmentation des phrases en mots comme la première étape nécessaire à l'analyse en dépendances. L'évaluation des méthodes d'analyse en dépendances s'effectuant à l'accoutumée sur des phrases pré-découpées et non sur des textes, nous ne tiendrons pas compte d'une éventuelle étape de segmentation des textes en phrases. Nous employons ici le terme **mot** pour toute forme de segmentation : les mots simples ou les expressions multi-mots (e.g. les mots composés). La segmentation d'une phrase W consiste à trouver une segmentation $W = w_1...w_n$ où les w_i avec $i \in [1,n]$ sont des mots et n le nombre de mots. En outre, les enjeux de la segmentation sont différents selon le point de vue duquel on se place :

- un point de vue plutôt linguistique, qui met en avant le sens et donc privilégie le regroupement des mots simples en expressions multi-mots si, pour la phrase, le sens du tout est plus pertinent que le sens de chacun des mots séparément. Il s'agira par exemple de conserver en un seul segment de texte les noms d'institutions et d'entreprises tels que « Assemblée Nationale » ou « Mer morte » ;

- un point de vue plus informatique, qui privilégie l'efficacité en découpant la phrase en mots simples d'abord en ne se préoccupant pas de possibles regroupements quitte à effectuer ses réunions lors d'étapes ultérieures. En ce cas les exemples précédents seraient divisés chacun en deux mots distincts, respectivement de catégories grammaticales *nom* et *adjectif* ;

La segmentation de base, mécanique, peut-être effectuée à partir du découpage de la phrase en mots simples suivant les espaces, les ponctuations, les apostrophes mais tenant compte des cas particuliers (e.g. les mots tels que « aujourd'hui », les nombres comportant des virgules ou des espaces). L'aide d'un lexique peut-être intéressante pour capturer les expressions multi-mots tels que les expressions figées (e.g. « carte blanche ») ou les mots composés. Cependant, s'appliquer à segmenter correctement ces unités de sens peut rapidement introduire des ambiguïtés, des erreurs et induire un traitement hétérogène de ces cas particuliers. Les traitements statistiques sont alors utiles pour atténuer les ambiguïtés en s'appuyant sur le contexte de la phrase pour déterminer la probabilité d'un regroupement entre plusieurs mots simples. Des questions encore plus approfondies sur la segmentation peuvent mener, par exemple, à des travaux sur la reconnaissance des entités nommées. La tâche de segmentation des phrases en mots peut alors devenir une étape difficile nécessitant l'apport de nouvelles informations et un travail approfondi. De plus, en fonction de la langue et des besoins, la tâche de segmentation est souvent corrélée à la tâche d'étiquetage grammatical.

Pour une phrase $W = w_1...w_n$ de taille n la tâche d'étiquetage grammatical ou étiquetage morpho-syntaxique consiste à trouver la séquence d'étiquettes $T = t_1...t_n$ telle que, pour chaque $i \in \{1,n\}$, t_i est l'étiquette grammaticale assignée au mot w_i. Les étiquettes grammaticales, appelées aussi parties du discours, désignent les fonctions grammaticales occupées par les mots dans la phrase (e.g. verbe, déterminant, adjectif). Il s'agit donc de classer les mots des phrases par catégories. Cette étape d'étiquetage grammatical, couramment désignée par son équivalent anglais *Part-Of-Speech (POS) tagging*, est effectuée dans de multiples travaux du domaine du traitement de la langue. Du fait des très bons scores obtenus par cette tâche (plus de 97% de précision pour le français dans les travaux standards), elle est particulièrement utile dans la désambiguïsation de l'analyse syntaxique. Cependant, il a été montré que le jeu d'étiquettes (ensemble des étiquettes grammaticales préalablement défini) utilisé pour la tâche d'étiquetage grammatical, réalisé en amont d'une analyse syntaxique, avait un impact sur les performances de ces analyses (Crabbé et Candito, 2008). Il est donc important de bien choisir le jeu d'étiquettes dans l'objectif de maximiser simultanément les scores d'étiquetage grammatical et d'analyse syntaxique. Cependant, dans

le cas des méthodes dirigées par les données, les jeux d'étiquettes assignés par les étiqueteurs proviennent des corpus utilisés pour l'entraînement et sont donc dépendant des données disponibles. Par exemple, la majorité des étiqueteurs et des travaux d'étiquetage grammatical pour le français exploite les données du *French Treebank*. Ce corpus comporte deux niveaux d'étiquetage. Le premier, comportant 13 étiquettes (catégories) grammaticales, peut être considéré comme trop général et donc pas assez précis alors que les sous-catégories[3] du second niveaux (correspondant à un jeu de 34 étiquettes) sont beaucoup plus précises. La trop grande diversité d'un jeu d'étiquettes peut rendre compliqué la distinction entre des étiquettes proches même pour les annotateurs humains (entraînant par exemple des inconsistances entre les annotations si plusieurs annotateurs sont impliqués (Boudin et Hernandez, 2012)). Les travaux de Crabbé et Candito (2008) ont permis de définir un nouveau jeu d'étiquettes se situant entre les deux niveaux et comportant 29 étiquettes correspondant parfois à une combinaison entre une catégorie supérieure et une catégorie inférieure. Ce jeu d'étiquettes particulier a alors été adopté pour l'étiqueteur Melt (Denis et Sagot, 2009, 2012) et est devenu le jeu d'étiquettes standard pour le français.

Parmi les méthodes d'étiquetage grammatical on trouve tout d'abord celles basées sur les règles (*rule-based*). Une des applications de cette méthode fut l'utilisation du système TAGGIT (Greene et Rubin, 1971) dans le processus d'annotation du corpus BROWN (Francis et Kucera, 1982). La méthode est divisée en deux étapes. La première consiste à étiqueter les mots non-ambigus (ayant une seule étiquette grammaticale possible) à l'aide d'un dictionnaire, puis d'étiqueter les mots non-étiquetés par des étiquettes fréquentes telles que *Verbe* ou *Nom*. La seconde étape consiste à modifier les étiquettes des mots dont le contexte est sûr (i.e. dont les étiquettes des mots voisins sont non-ambiguës) grâce à des règles décrites manuellement. Cette première méthode peut être vue comme contraignante du fait de l'écriture manuelle des règles. Par la suite, Brill (1995) proposa une méthode similaire dans laquelle les règles sont générées à partir de patrons de règles. Ce travail est à l'origine des méthodes fondées sur un système de transformation (*transformation-based learning* (TBL)). La première étape, semblable à celle de la méthode précédente, donne à chaque mot une première étiquette. Puis le système apprend des règles de correction à partir des données correctement annotées. Un ensemble de patrons de règles, accordant l'inclusion de conditions, de noms d'étiquettes à modifier, permet de générer ces règles. Parmi ces règles, celles engendrant le moins d'erreurs sur l'ensemble des données sont conservées. On obtient donc ici un premier système d'étiquetage incluant une étape de correction dirigée par les données.

Par la suite, les réseaux bayésiens ont pris une grande place dans le domaine de l'étiquetage grammatical, en commençant par les modèles de Markov cachés (techniquement équivalents aux automates finis probabilistes).

Définition 12 *Modèle de Markov caché (*Hidden Markov Model *(HMM))*
Soit un HMM $\Lambda = (A,B,\Pi)$ *avec :*

- n *le nombre d'états du HMM ;*
- $S = \{s_1,...s_n\}$ *les états ;*
- M *le nombre de symboles observables, i.e. la taille de l'alphabet ;*
- $V = \{v_1,...,v_M\}$ *les observations, i.e. les symboles de l'alphabet ;*

et tel que :

[3] Les sous-catégories correspondent à des divisions des catégories grammaticales, e.g. la catégorie V (les verbes) est divisée en 6 sous-catégories (V,VIMP,VINF,VS,VPP et VPR).

- *A est une matrice de probabilités de transitions avec $a_{ij} = A(i,j) = P(q_{t+1} = s_j \mid q_t = s_i)$ la probabilité de transition de s_i à s_j ;*
- *B est une matrice de probabilités d'observations avec $b_j(k) = P(O_t = v_k \mid q_t = s_j)$ la probabilité d'observer v_k lorsque l'état courant est s_j ;*
- *Π est le vecteur des probabilités initiales où $\Pi_i = P(q_1 = s_i)$.*

L'apprentissage des paramètres d'un HMM consiste à maximiser

$$P(\Theta \mid \Lambda) = \prod_{O \in \Theta} P(O \mid \Lambda)$$

la probabilité d'un ensemble Θ de séquences d'observations $O = O_1 ... O_T$ (i.e. l'ensemble des phrases du corpus d'apprentissage) sachant Λ. L'algorithme d'apprentissage, Baum-Welch (Baum, 1972), procède par ré-estimation des paramètres à partir de chaque séquence d'observations. Cette étape d'apprentissage du modèle statistique permet ensuite d'effectuer l'étape d'étiquetage attendue dans le cadre de l'étiquetage grammatical. Il s'agit, dans le domaine de l'apprentissage automatique, du problème de recherche du chemin optimal dans un automate probabiliste. L'idée est de trouver la suite d'états la plus probable pour la séquence d'observation O correspondant à la phrase donnée. À la suite d'états on fait alors correspondre la suite des étiquettes grammaticales associées aux mots. Pour effectuer cette étape, c'est l'algorithme de Viterbi (1967) qui est employé.

Parmi les étiqueteurs exploitant les capacités des HMM, on trouve le TnTtagger de Brants (2000). D'autre part, citons le TreeTagger de Schmid (1994) employant une méthode un peu différente, proche des N-grammes.

Parmi les modèles graphiques probabilistes, les modèles d'entropie maximum de Markov (*Maximum Entropy Markov Model* (MEMM)) constituent une alternative aux HMM. Contrairement aux HMM, les MEMM sont capables de tenir compte d'une diversité plus importante de traits provenant du contexte du symbole observé. Techniquement, la probabilité d'une transition $s \rightarrow s'$ sachant une observation O_t est calculée suivant le même modèle que dans les méthodes des classificateurs MaxEnt (maximum d'entropie) :

$$P(q_{t+1} = s' \mid q_t = s, o_t = v) = \frac{1}{Z(o_t, s)} exp\left(\sum_a \lambda_a f_a(o_t, s)\right)$$

où $f_a(o,s)$ renvoie la valeur d'un trait f_a pour une observation o_t en s, λ_a est un paramètre (poids) estimé lors de la phase d'apprentissage et $Z(o,s')$ est une variable de normalisation. L'inconvénient principal des MEMM est connu sous le nom *label bias problem* en anglais. Cela signifie que, lors de la recherche du chemin optimal, les états ayant le plus faible nombre de transitions entrantes seront favorisés même par rapport à ceux ayant une plus forte probabilité locale. Les CRF, appartenant également à la classe des modèles graphiques probabilistes, sont renforcés par rapport aux MEMM pour ne pas subir les désavantages du *label bias problem*. Pour le français, citons les travaux conjoints de chercheurs du LIFO et du LIGM (Constant *et al.*, 2011) utilisant la chaîne de traitement SEM ou le segmenteur-étiqueteur LGTagger avec exploitation de ressources telles que Lefff (Sagot, 2010) ou DELA (Courtois, 1990).

D'autres travaux présentent des techniques associant des informations lexicales externes à des méthodes classiques d'étiquetage. Un des outils les plus connus dans le domaine de l'étiquetage grammatical pour le français utilisant le Lefff (Sagot, 2010) est le Melt tagger (Denis et Sagot, 2009, 2012). Il utilise par ailleurs la méthode MaxEnt pour effectuer l'étiquetage.

Les classificateurs, dont la méthode MaxEnt, sont aussi des méthodes couramment utilisées dans le domaine de l'étiquetage grammatical. Une différence avec les réseaux bayésiens est due

au fait que les classificateurs étiquettent indépendamment chaque observation tandis que les réseaux bayésiens étiquettent une séquence d'observation. Les classificateurs maximisent la probabilité d'une étiquette pour un mot donné alors que les réseaux bayésiens cherchent à maximiser la probabilité de la séquence d'étiquettes pour la phrase donnée. Avec les classificateurs, il est donc possible d'obtenir une liste d'étiquettes, pour chaque mot, ordonnées suivant leurs probabilités. Parmi les outils et méthodes développés pour réaliser l'étiquetage grammatical de phrases on trouve également :

- le Standford Tagger (Toutanova et Manning, 2000) exploitant les avantages de la méthode MaxEnt ;
- une méthode améliorée de Collins (2002) basée sur la version *Averaged Perceptron* de l'algorithme perceptron ;
- le SVMTool de Giménez et Márquez (2004).

3.3.2 Supertagging

Dans le domaine de l'analyse syntaxique, les pré-traitements ne se limitent pas à la segmentation et à l'étiquetage grammatical. Le supertagging, bien que décrit comme « presque de l'analyse syntaxique » (*almost parsing*) par Bangalore et Joshi (1999), peut-être classé parmi les pré-traitements utiles à l'analyse syntaxique, en particulier aux analyses basées sur les grammaires. À l'origine, le supertagging fut proposé pour la désambiguïsation des analyses avec les grammaires LTAGs (Schabes, 1990). Les LTAGs sont des grammaires dont les membres droits des règles sont des arbres élémentaires possédant au moins une feuille étant un mot (voir section 2.3 chapitre 2). Le supertagging consiste alors à trouver, pour chaque mot d'une phrase donnée, la liste des arbres élémentaires contenant ce mot. L'analyse syntaxique consiste ensuite à combiner ces arbres (ce qui correspond aux opérations de substitution et d'adjonction dans le cas des LTAGs) pour trouver les différentes analyses possibles (les arbres de dérivations) pour la phrase donnée. Dans le cas du supertagging, ces arbres seront appelés des supertags, comme illustré par la figure 3.1.

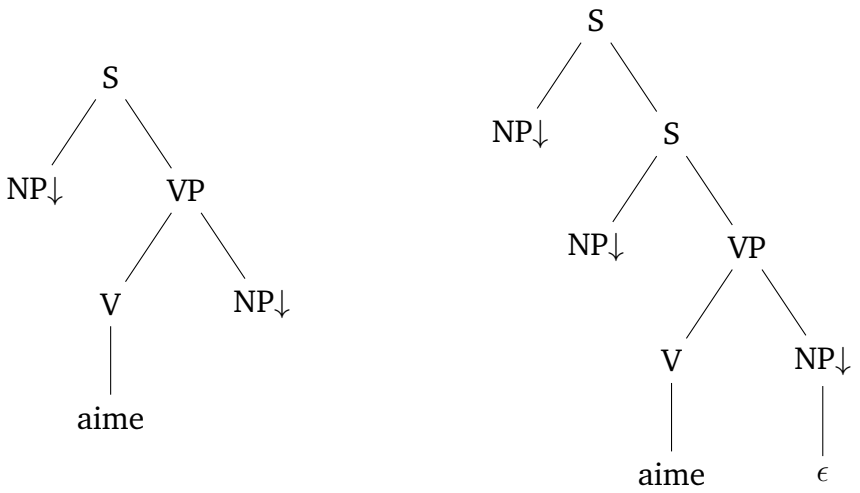


FIGURE 3.1 – Supertags (arbres élémentaires) pour le mot « aime ». Les noeuds NP↓ sont des noeuds substituables par d'autres arbres élémentaires dont le noeud racine est NP↑.

L'étape de supertagging peut être utilisée pour restreindre plus ou moins le nombre de supertags assignés à chaque mot. Le choix de la quantité de supertags préservés et fournis à l'étape

d'analyse syntaxique ultérieure dépend de l'objectif poursuivi. Restreindre le nombre de supertags par mot permet de réduire la complexité des analyses syntaxiques dirigées par les grammaires mais peut parallèlement réduire la précision des analyses si les supertags sélectionnés sont incohérents les uns par rapport aux autres. Le gain apporté par le supertagging dans ce genre d'analyses a été étudié par Sarkar (2010) dans le cas des LTAGs mais également par Clark et Curran (2004) pour l'analyse avec des CCGs (*Combinatory Categorial Grammars*).

L'inclusion de modèles statistiques dans la mise en œuvre d'une étape de supertagging a été proposée dans différents travaux. En effet, les méthodes dirigées par les données sont bien adaptée à leur application car elles permettent de réduire l'ambiguïté des analyses en restreignant les listes de supertags possibles selon leurs probabilités. Dans ce but, notons les travaux de Harbusch *et al.* (2010) proposant une méthode de supertagging à base de HMM. En outre, les classificateurs sont aussi particulièrement employés pour ce genre de tâche. Notons le travail de Shen (2010), exploitant le potentiel d'un classificateur multi-classe SNoW (Roth, 1998) ou celui de Chen (2001) utilisant la méthode MaxEnt. Néanmoins, les méthodes dirigées par les données peuvent engendrer l'oubli de supertags peu fréquents mais pourtant nécessaire à l'analyse. Dans ce cas ces méthodes dirigées par les données ne sont pas toujours exploitables. Par exemple, Boullier (2010) réduit une LTAG à un sur-ensemble régulier (automate à états finis) dans le but de conserver toutes les analyses possibles pour une phrase et d'obtenir un rappel de 100%.

L'idée première de Bangalore et Joshi (1999) était de prédire non pas des informations simples telles que les étiquettes grammaticales des mots mais des structures complexes permettant de capturer le contexte précis des mots. C'est ce qu'ils appellent la méthode « localement compliquée, globalement simplifiée » (CLSG pour « *complicate locally, simplify globally* ») car simplifiant de surcroît l'analyse syntaxique. Cependant, l'idée globale du supertagging, entre étiquetage et analyse, est de fournir des informations syntaxiques utiles pour la désambiguïsation des analyses syntaxiques et peut donc être adaptée à différents types de travaux. Le supertagging a donc été exploité par ailleurs pour l'analyse de différentes langues à travers différents formalismes et bien sûr pour l'analyse en dépendances. Par exemple, pour l'anglais, Nasr et Rambow (2010) comparent les performances d'un supertagger basé sur les HMM et d'un supertagger basé sur la méthode MaxEnt, pour l'analyse en dépendances. Nous citerons également le travail de Moot (2010) pour l'analyse en dépendances du néerlandais.

3.4 Analyses syntaxiques dirigées par les données

Dans cette section nous nous intéressons aux méthodes dirigées par les données pour l'analyse syntaxique. Nous revenons pour cela sur les méthodes présentées dans la section 2.3 du chapitre 2. Nous y distinguons alors les analyses que nous appelons semi-dirigées par les données des analyses entièrement dirigées par les données. Les analyses semi-dirigées par les données sont les analyses basées sur les grammaires. Ce qui nous conduit à l'étude des grammaires probabilistes qui associent règles syntaxiques et probabilités apprises à partir des données d'entraînement. D'autre part, les analyses entièrement dirigées par les données comprennent les systèmes d'analyses en dépendances, tels que l'analyse par satisfaction de contraintes et l'analyse par transition, dont les mécanismes sont totalement dépendants des données d'apprentissage.

3.4.1 Méthodes semi-dirigées par les données

Les méthodes d'analyse basées sur les grammaires produisent fréquemment, pour une phrase donnée, non pas une unique analyse mais un ensemble d'analyses. Il s'agit d'obtenir toutes les analyses

possibles pour cette phrase donnée. Pour une phrase syntaxiquement correcte vis-à-vis de la grammaire utilisée, le rappel est donc de 100% (en considérant que toute analyse termine) mais la bonne analyse est fréquemment noyée parmi les nombreuses possibilités d'analyse non ordonnées. Par ailleurs, le temps d'analyse des analyses basées sur les grammaires est habituellement trop élevé pour finalement parvenir à une analyse robuste et précise.

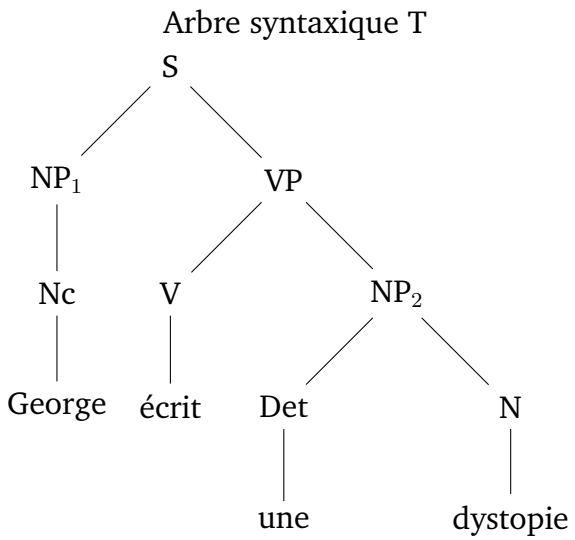
Le problème de fond à résoudre est l'ordonnancement des analyses. Le but n'est pas seulement d'avoir la bonne analyse parmi les sorties de l'analyseur mais d'obtenir la bonne analyse comme première sortie de l'analyseur. Il est donc nécessaire d'ordonner les analyses en leur attribuant des poids calculés à partir de différents critères. Ces critères peuvent provenir de différentes données e.g. les mots de la phrase, les règles de la grammaire utilisée. Notons ici la distinction que l'on fera entre deux méthodes d'analyses basées sur les grammaires : celles dans lesquelles les méthodes probabilistes sont intégrées et celles pour lesquelles les méthodes probabilistes sont appliquées en aval de l'analyse (post-traitement des analyses). En outre, un second objectif peut être, comme dans le cas du supertagging, la désambiguïsation des analyses syntaxiques dirigées par les données. Dans ce cas le but est également de restreindre les possibilités d'analyses en favorisant certaines analyses dans le but de réduire les temps de calcul tout en maximisant la probabilité d'obtenir la bonne analyse parmi les analyses générées.

Nous présentons, en premier lieu, les grammaires probabilistes. Pour illustrer, dans un cadre général, l'utilisation des grammaires probabilistes, nous nous limitons ici à la présentation des PCFG (*Probabilistic Context Free Grammar*) ou grammaires hors-contexte probabilistes et des algorithmes d'analyse qui y sont associés. Les PCFG sont des grammaires hors-contexte telles que décrites dans la section 2.3 du chapitre 2 et pour lesquelles chaque règle est combinée à une probabilité conditionnelle. Ainsi, pour une PCFG $G = \langle N,\Sigma,\Pi,S,D\rangle$ où D est une fonction de distribution des probabilités, la somme des probabilités des règles de la forme $A \rightarrow \alpha$, avec A fixé $\in\ N$ et $\alpha \in (\Sigma \cup N)^*$, est égale à 1. Le but d'une analyse avec une PCFG est alors de trouver l'analyse T (i.e. un arbre) pour une phrase W telle que la probabilité $P(T,W)$ est maximale et

$$P(T,W) = \prod_{n \in T} P(R(n))$$

où R est une règle de la grammaire et n un noeud de l'arbre d'analyse. Un exemple de calcul de la probabilité d'un arbre syntaxique est donné par la figure 3.2. Pour parvenir à estimer les analyses les plus probables, différents algorithmes dérivés des algorithmes classiques d'analyse des CFG ont été proposés. Jurafsky et Martin (2000) présentent une version probabiliste de l'algorithme CYK, adaptée de Collins (2003) et Aho et Ullman (1972). Citons, de plus, la version probabiliste de l'algorithme d'Earley (Stolcke, 1995).

Étant donné l'aspect statistique des PCFG, l'intégration d'une phase d'apprentissage dans la construction de ces grammaires est possible. Le processus d'apprentissage se fait de deux manières différentes : à partir d'une grammaire préalablement construite où uniquement à partir d'un ensemble de données d'apprentissage non accompagné d'une grammaire. Si il n'y a aucune grammaire CFG pré-existante, la construction d'une PCFG consiste en une technique d'extraction de grammaire combinée à l'apprentissage des probabilités des règles. Dans le cas où une grammaire CFG est pré-existante, il est seulement nécessaire d'apprendre les probabilités des règles. L'apprentissage des probabilités, à partir d'un corpus correctement annoté, consiste à compter le nombre de fois qu'une règle est utilisée durant l'analyse des phrases du corpus et à normaliser ce résultat en fonction du nombre de règles ayant la même partie gauche (i.e. le nombre de règles étant de la forme $A \rightarrow \alpha$). Cette méthode, nécessitant l'exploitation d'un corpus correctement annoté, fait donc partie des méthodes supervisées. Néanmoins, il existe des algorithmes permettant d'obtenir les probabilités des PCFG à partir d'ensembles de phrases brutes. L'algorithme de Baker (2005), appelé *Inside-Outside algorithm*, en est un exemple. Les phrases sont préalablement analysées, puis



Règles

R(S) : $S \rightarrow NP_1 \ VP$
R(NP_1) : $NP \rightarrow Nc$
R(VP) : $VP \rightarrow V \ NP$
R(NP_2) : $NP \rightarrow Det \ N$

P(T,W) = R(S)*R(NP_1)*R(VP)*R(NP_2)

FIGURE 3.2 – Probabilité P(T,W) de l'arbre syntaxique T pour la phrase W=« George écrit une dystopie » à partir de la probabilité des règles employées lors de l'analyse.

les analyses, potentiellement ambiguës, sont utilisées par l'algorithme pour déterminer les probabilités des règles.

Parmi les grammaires que nous avions présentées dans la section 2.3 du chapitre 2, il existe des versions probabilistes des grammaires TAG (Resnik, 1992; Schabes, 1992) et de la *Link Grammar* (Lafferty *et al.*, 1992). Par ailleurs, on retrouve la notion de modèle probabiliste dans les *Weighted Bilexical Grammars* (WBG) d'Eisner (1997) permettant de réaliser des analyses en dépendances à partir d'automates finis déterministes. Dans le domaine de l'analyse en dépendances, notons de plus les travaux de Wang et Harper (2004) sur les *Constraint Dependency Grammars*.

Le problème d'ordonnancement des analyses peut aussi être vu comme un post-traitement appliqué aux sorties d'un analyseur (Charniak et Johnson, 2005). Cette tâche permet d'ordonner la liste des arbres produits par les analyseurs basés sur les grammaires, autant pour les grammaires probabilistes (Collins et Koo, 2005) que pour les grammaires non-probabilistes, et ainsi d'améliorer la précision des analyses. Cependant, le ré-ordonnancement des arbres après analyse présuppose que l'analyseur fournit au moins une sortie. Or, les analyseurs basés sur les grammaires (n'incluant pas de pré-traitements tel que le supertagging) cherchent toutes les analyses possibles et par conséquent ne sont pas toujours capables de produire une sortie du fait de la croissance exponentielle de l'espace de recherche et donc d'une trop grande complexité de l'analyse et d'un temps d'analyse trop élevé. Dans le but de réduire la complexité des analyses tout en assurant une bonne précision, Huang (2008) propose d'effectuer un ré-ordonnancement des forêts d'analyses où une forêt d'analyse est une représentation compacte de tous les arbres de dérivations d'une analyse pour une phrase donnée et à partir d'une grammaire hors-contexte. Cette méthode de ré-ordonnancement a aussi été exploitée par Clark et Curran (2007) dans des travaux dédiés aux CCGs (*Combinatory Categorial Grammars*).

3.4.2 Méthodes dirigées par les données

Les méthodes totalement dirigées par les données ont pris une grande importance parmi les travaux récents, en particulier dans le domaine de l'analyse en dépendances. Les formalismes d'analyse présentés dans la section 2.3 du chapitre 2 associés à des méthodes d'apprentissage automatique permettent d'effectuer des analyses déterministes et efficaces. Les temps d'analyse sont réduits et les évaluations de ces analyseurs révèlent des scores très intéressants si des corpus correctement annotés et suffisamment grands sont disponibles pour l'entraînement.

Nous avons pu voir que le principe d'une analyse en dépendances par satisfaction de contraintes est de trouver l'arbre couvrant maximal d'un graphe pour une phrase donnée. La méthode d'analyse permettant de trouver cet arbre est donnée par l'algorithme Chu-Liu-Edmonds qui fait disparaître les cycles du graphe en sauvegardant uniquement les dépendances favorisant le score de l'arbre couvrant. Il est donc nécessaire de fournir un graphe pondéré en entrée de l'algorithme. La première étape de l'analyse en dépendances par satisfaction de contraintes est donc la construction d'un graphe connexe potentiellement complet dont chaque arc est valué par un poids représentant la probabilité que les mots formant les sommets de l'arc soient engagés dans une relation de dépendance (étiquetée ou non). Le poids d'un arc est calculé à partir du produit entre un vecteur de traits $f(w_i,l,w_j)$ pour une dépendance (i,l,j) et un vecteur de poids fixé. Le vecteur de traits décrit les caractéristiques (il s'agit habituellement de traits binaires) de la dépendance tandis que le vecteur de poids définit l'importance de chacun des traits à partir de l'apprentissage sur un ensemble de données correctement annotées. Dans les travaux de Mcdonald *et al.* (2005), le vecteur de poids est appris grâce à une adaptation de l'algorithme MIRA (*Margin Infused Relaxed Algorithm*) (Crammer et Singer, 2003) , une méthode de classification linéaire multi-classes incrémentale (i.e. le vecteur de poids est recalculé par l'algorithme après le traitement de chaque phrase/graphe de dépendances de l'ensemble d'apprentissage).

D'autre part, l'analyse par transition déterministe consiste à rechercher la séquence de transitions permettant de construire une structure de dépendances correcte pour une phrase donnée en optimisant localement la prédiction des transitions. C'est à dire que pour chaque nouvelle configuration le système doit être capable de prédire la transition la plus probable. Un tel système, utilisant les transitions du formalisme projectif d'origine, permet de construire une structure de dépendances en 2n-1 étapes pour une phrase donnée de taille n (Nivre, 2010). Le déterminisme du système résulte de l'emploi de méthodes de classification pour la prédiction des transitions. Pour chaque nouvelle configuration créée, une et une seule transition est prédite. À chaque couple configuration/transition doit donc pouvoir être associé un poids représentant la probabilité que la transition soit appliquée à la configuration courante (l'état courant de l'automate). Un poids est calculé à partir d'un vecteur de traits décrivant la configuration (comprenant des informations sur les mots traités, les dépendances déjà assignées, etc.). L'apprentissage des poids se fait alors généralement grâce à une méthode de classification et à partir d'ensembles de données (graphes de dépendances) préalablement traduits en séquences de transitions. Ainsi, pour chaque configuration, la transition la plus probable est prédite. À travers les multiples travaux sur l'analyse par transition, on constate que les classificateurs linéaires on été largement exploités. On trouve ainsi des travaux utilisant par exemple les SVM (Kudo et Matsumoto, 2002; Yamada et Matsumoto, 2003; Nivre *et al.*, 2006b), la méthode MaxEnt (Cheng *et al.*, 2005b) ou la méthode *Memory-based learning* (Nivre *et al.*, 2004; Attardi, 2006).

Les méthodes d'analyse par satisfaction de contraintes et par transitions sont déterministes de part leur objectif qui est de trouver une seule structure de dépendances de plus haut score pour une phrase donnée. Cependant, tout comme les analyses basées sur les grammaires, ces méthodes peuvent être employées pour produire la liste des structures de dépendances les plus probables pour la phrase. Cet axe de recherche a fait l'objet d'études récentes, en particulier à travers des travaux manipulant la recherche en faisceau (*beam-search*). La recherche en faisceau

est une optimisation de l'algorithme de parcours en profondeur dans un graphe, conservant un nombre prédéterminé de solutions. Ce nombre est désigné comme étant la largeur du faisceau. Les méthodes déterministes vues précédemment peuvent alors être vues comme des méthodes pour lesquelles une largeur de faisceau de 1 fut appliquée. La recherche en faisceau permet de réduire l'espace de recherche dans le graphe en éliminant les chemins les moins probables. Par conséquent, cette technique doit être combinée à une méthode permettant d'évaluer les scores de probabilités des chemins possibles dans le graphe. Les classificateurs sont encore une fois très appréciés pour effectuer ce genre de tâche. Par exemple, Zhang et Clark (2008, 2011) intègrent la recherche en faisceau combinée au Perceptron dans les méthodes d'analyse par satisfaction de contraintes et par transition. En ce qui concerne le français, Urieli (2013) propose à travers l'outil *Talisman* une chaîne de traitement complète pour l'analyse en dépendances du français incluant la recherche en faisceau à chaque étape : segmentation, étiquetage grammatical, analyse en dépendances.

3.5 Conclusion

Aujourd'hui, la disponibilité des corpus en dépendances et l'efficacité des techniques d'apprentissage automatique rendent de plus en plus intéressante l'utilisation de méthodes dirigées par les données dans les processus d'analyse syntaxique.

Notons tout d'abord l'importance qu'ont prises les méthodes dirigées par les données dans les étapes de pré-traitement tel que l'étiquetage grammatical. Du fait du nombre important de données annotées et de la faible complexité de la tâche (étiquetage d'une information simple), les méthodes dirigées par les données obtiennent des scores importants dans ce domaine et sont devenues omniprésentes.

Dans le domaine de l'analyse syntaxique, ces méthodes se sont intégrées, tout d'abord, aux méthodes existantes basées sur les grammaires. Les méthodes d'analyses basées sur les grammaires, étant inscrites dans une longue tradition linguistique, ont considérablement évolué avec l'inclusion de méthodes dirigées par les données. La première étape fut l'évolution des grammaires classiques en grammaires probabilistes. Puis, des étapes de pré-traitement (supertagging) et de post-traitement (ré-ordonnancement) ont émergé, renforçant la précision et la rapidité des méthodes d'analyse basées sur les grammaires.

Par ailleurs, l'analyse en dépendances est aujourd'hui fortement influencée par les méthodes entièrement dirigées par les données telles que l'analyse par satisfaction de contraintes et l'analyse par transition. L'attrait pour ces méthodes est du à leur rapidité d'analyse et leur précision.

Les méthodes dirigées par les données sont un fil conducteur dans le développement de nos travaux de thèse. Nous les exploitons tant pour l'analyse en dépendances basée sur les grammaires catégorielles de dépendances que pour l'analyse en dépendances par transition.

4

Les grammaires catégorielles de dépendances (CDG)

4.1 Introduction

Dans le chapitre précédent, nous présentions une vue globale sur les différentes méthodes dirigées pas les données dans le domaine de l'analyse syntaxique et nous évoquions le fait qu'elles seraient un fil conducteur tout au long de nos travaux. Dans ce chapitre, nous présentons le formalisme qui est au cœur de nos travaux : les grammaires catégorielles de dépendances ou CDG (pour *Categorial Dependency Grammar*) (Dekhtyar et Dikovsky, 2004, 2008). En effet, nos premiers travaux nous ont amené à travailler d'une part sur des étapes de pré-traitement nécessaires à l'évolution d'un analyseur en dépendances pour les grammaires catégorielles de dépendances. D'autre part, nos seconds travaux sur l'analyse par transition ont été fortement inspirés par le formalisme des grammaires catégorielles de dépendances.

Les premiers travaux posant les bases de la représentation en dépendances (Tesnière, 1959; Mel'cuk, 1988) proposaient des structures syntaxiques dans lesquelles les cas de non-projectivité (voir la section 2.2 du chapitre 2) étaient naturellement inclus car intrinsèques à la langue. La question de la non-projectivité n'était, à ce moment là, pas posée d'un point de vue informatique, en terme de complexité des systèmes générant de telles structures, mais d'un point de vue linguistique, mettant clairement en avant la nécessité de ne pas restreindre les représentations en dépendances à des structures projectives. Les principes linguistiques des grammaires catégorielles de dépendances trouvent leurs origines dans la théorie Sens-Texte de Mel'cuk (1988). Elles furent développées dans le but de pouvoir décrire la syntaxe des langues naturelles sans restriction sur la projectivité.

Dans ce chapitre, nous présentons tout d'abord la représentation en dépendances induite par les grammaires catégorielles de dépendances, incluant la gestion des dépendances non-projectives, et la terminologie particulière associée à cette représentation. Dans la section qui suit, nous rappelons le formalisme général des grammaires catégorielles classiques pour parvenir ensuite à la description des grammaires catégorielles de dépendances. Les grammaires catégorielles de dépendances incluent les principes des grammaires catégorielles pour décrire les dépendances projectives et étendent leur description aux dépendances non-projectives par l'intégration de nouveaux éléments

et règles de calcul. Les CDG sont alors plus expressives que des grammaires hors-contexte classiques qui ne sont pas adaptées à la gestion des dépendances non-projectives. Par ailleurs, dans le cadre des CDG et de l'analyse en dépendances du français, nous présentons en particulier la grammaire catégorielle de dépendances du français (Dikovsky, 2011) développée parallèlement à un corpus en dépendances du français. Les cas de non-projectivité sont présents en nombre suffisant dans la langue française pour impacter les résultats des analyses en dépendances. De plus, leur prise en compte dans la description de la syntaxe de la langue et dans les corpus est pertinente car la bonne gestion de ces dépendances peut être déterminante lors de certains traitements du langage naturel. Pour finir, nous décrivons l'environnement CDG Lab comprenant l'analyseur des CDG ainsi que les grammaires catégorielles de dépendances et les corpus en dépendances ayant été développés à l'aide de l'outil d'annotation de l'analyseur.

4.2 Représentation et terminologie

Un format classique de représentation en dépendances (tel qu'employé par les campagnes d'évaluation CoNLL, voir la sous-section 2.4.3 du chaptire 2) admet une unique dépendance entrante pour tout mot d'une phrase donnée. La représentation en dépendances sur laquelle se basent nos travaux n'est pas standard dans le sens où elle inclue des informations supplémentaires. En effet, dans cette représentation, que nous désignons par le terme **structure de dépendances**, chaque mot admet une unique dépendance projective mais peut également admettre une dépendance non-projective. Cette représentation est induite par le formalisme des grammaires catégorielles de dépendances.

Une structure de dépendances est un graphe orienté acyclique dont les nœuds sont les mots[1] de la phrase à analyser et les arcs représentent les dépendances reliant ces mots. Dans un tel graphe, une dépendance est un arc orienté et est étiquetée par un nom de dépendance. On désigne une dépendance par (i, d, j) où i et j sont les indices des mots impliqués dans la dépendance (w_i est le gouverneur de la dépendance et w_j est le subordonné) et d est une étiquette, i.e. le nom de la dépendance caractérisant la relation syntaxique liant les deux mots. En outre, comme dans le format de représentation standard, l'ordre dans lequel les mots apparaissent dans la phrase est conservé dans la structure de dépendances. Cet ordonnancement établit une relation d'ordre strict $<$ entre les mots de la phrase. Formellement, $w_i < w_j$ signifie que w_i précède w_j dans la phrase. Par conséquent, si $w_i < w_j$ alors (i, d, j) est une dépendance droite (le dépendant est à droite) tandis que si $w_j < w_i$ alors (i, d, j) est une dépendance gauche (le dépendant est à gauche). Une relation de dépendances peut également être décrite par une relation de dominance directe telle que décrite dans la section 2.2 du chapitre 2. Par exemple, $w_i \xrightarrow{d} w_j$ indique la dominance directe de w_j par w_i et ainsi la dépendance étiquetée d entre ces deux mots. Un exemple de structure de dépendances simple est donné par la figure 4.1.

Par ailleurs, la conservation de l'ordre des mots induit parfois la non-projectivité des structures de dépendances. Nous rappelons (voir la sous-section 2.2.2 du chapitre 2) qu'une dépendance est non-projective si au moins un mot situé entre le gouverneur et le subordonné de la dépendance ne dépend pas, directement ou indirectement, du gouverneur. Conséquemment, les dépendances peuvent se croiser dans la représentation. Dans la représentation en dépendances induite par les CDG, chaque dépendance non-projective est couplée à une dépendance projective de même rôle syntaxique appelée **ancre**. Il y a donc trois sortes d'arcs distincts : les **dépendances projectives**, les **dépendances non-projectives** (ou **discontinues**) et les **ancres**. La distinction entre ces trois sortes

[1] Dans les grammaires catégorielles de dépendances les différentes formes des mots (tout segment de phrase lexicalement valide) sont nommées **unités lexicales**. Nous utilisons ici toujours le terme de **mot** pour désigner cet ensemble d'unités lexicales pouvant contenir des mots simples, mots composés ou expressions multi-mots.

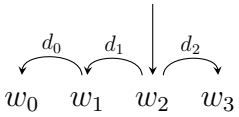


FIGURE 4.1 – Structure de dépendances D={V, A} pour la phrase W = $w_0w_1w_2w_3$, avec V = $\{w_0, w_1, w_2, w_3\}$ et A = $\{(1, d_0, 0),\ (2, d_1, 1),\ (2, d_2, 3)\}$. $(2, d_1, 1)$ est une dépendance gauche et $(2, d_2, 3)$ est une dépendance droite. Il n'existe pas de dépendance $(2,d',0)$, néanmoins w_2 domine w_0 par transitivité, i.e. $w_2 \xrightarrow{*} w_0$.

d'arcs est nécessaire pour pouvoir représenter dans une même structure les relations continues et discontinues des phrases. Les dépendances projectives sont des dépendances qui ne se croisent pas les unes avec les autres. Elles sont représentées par les arcs pleins. Les dépendances non-projectives sont les dépendances qui peuvent croiser les autres dépendances. Elles sont représentées par des arcs en pointillés. Et les ancres sont des dépendances permettant de lier localement une dépendance non-projective. Elles sont représentées par des arcs pleins situés en dessous de la phrase. La figure 4.2 illustre l'emploi des trois sortes d'arcs dans une seule structure de dépendances. Une structure de dépendances qui comprendrait uniquement des dépendances projectives et des ancres serait une structure de dépendances entièrement projective. Par ailleurs, une structure de dépendances sans ancres correspond à un format standard de représentation en dépendances. Nous désignons également la représentation induite par les CDG comme étant une représentation mixte projective/non-projective.

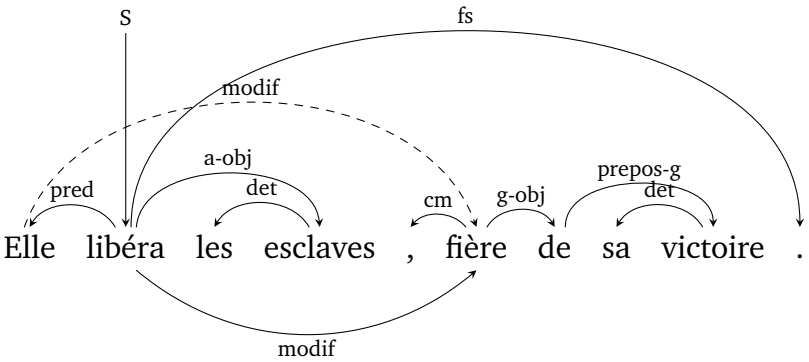


FIGURE 4.2 – Structure de dépendances non-projective pour la phrase « Elle libéra les esclaves, fière de sa victoire. ». La dépendance non-projective $(1,modif,6)$ (en pointillé) est couplée à une dépendance ancre $(2,modif,6)$ (en dessous).

4.3 Formalisme grammatical

4.3.1 Les grammaires catégorielles

Les grammaires catégorielles, aussi appelées grammaires AB, résultent à l'origine des travaux de Ajdukiewicz (1935) et de Bar-Hillel *et al.* (1964). Elles diffèrent des grammaires classiques (les grammaires hors-contexte, voir la sous-section 2.3.1 du chapitre 2) de part la description des symboles non-terminaux et des règles employées. En effet, il s'agit de types plutôt que de symboles

non-terminaux et non de règles de production mais de règles de réduction, ce qui change la manière d'aborder ces grammaires.

Définition 13 *Grammaire catégorielle*
Une grammaire catégorielle est définie par un quadruplet $G = \langle \Sigma, Tp, f, S \rangle$ *où :*

- Σ *est un ensemble de symboles terminaux, i.e. un alphabet ;*
- Tp *est un ensemble de types (ou catégories) construit à partir d'un ensemble de types primitifs* Pr *tel que :*
  - *si* $A \in Pr$ *alors* $A \in Tp$
  - *si* A,$B \in Tp$ *alors* $A\backslash B \in Tp$ *et* $A/B \in Tp$.
- f *est une fonction qui à chaque symbole terminal associe un sous-ensemble fini de types ;*
- $S \in Tp$ *est l'axiome.*

L'ensemble des types ne comprend pas seulement des catégories simples (types primitifs) mais également des catégories complexes décrivant le contexte grammatical des mots. Par exemple, dans une phrase telle que « La mixité est bénéfique », le déterminant « la » est décrit par le type primitif *Det* et le nom commun « mixité » est décrit par le type *Det*$\backslash$ *N* signifiant que le mot de classe *nom* attend un déterminant à gauche pour former un groupe. Ainsi, chaque symbole terminal (mot) $\alpha \in \Sigma$ peut être décrit par un ou plusieurs types (on note $\alpha \mapsto A$ avec $A \in Tp$) selon les contextes syntaxiques dans lesquels les mots apparaissent. Les grammaires catégorielles sont donc des grammaires fortement lexicalisées [2].

Par ailleurs, les grammaires catégorielles disposent d'un lien fort avec la logique de part les règles de dérivation qu'elles utilisent (proche de la déduction naturelle). Il s'agit de règles de réduction :

- à droite : $\forall A,B \in Tp,\ [A/B]\ B \vdash A$;
- et à gauche : $\forall A,B \in Tp,\ A\ [A\backslash B] \vdash B$.

Un exemple d'arbre de dérivation obtenu pour la phrase « La mixité est bénéfique » avec une grammaire catégorielle est donné par la figure 4.3.

Le langage $L(G)$ reconnu par une grammaire catégorielle G est :
$L(G) = \{w = w_0...w_n \in \Sigma^+ \mid \forall\, i \in \{1,...,n\}, \exists\, A_i \in Tp$ tel que $w_i \mapsto A_i$ et $A_1...A_n \vdash S\}$.

4.3.2 Types et valences dans les CDG

Les grammaires catégorielles de dépendances, introduites par Dikovsky (2004); Dekhtyar et Dikovsky (2008), sont avant tout des grammaires catégorielles. Elles partagent donc certains principes de base quant à la définition des types et leur assignation à des symboles terminaux, en particulier en ce qui concerne les dépendances projectives.

[2] Une grammaire lexicalisée est une grammaire dans laquelle chaque règle de production produit au moins un symbole terminal. Dans le cas des grammaires catégorielles, il n'y a pas de règles de production telles que définies pour les grammaires hors-contexte. Néanmoins, la description de la syntaxe d'une langue avec une grammaire catégorielle repose sur la description des types qui sont eux-mêmes dépendants du lexique car chaque mot est associé à un type.

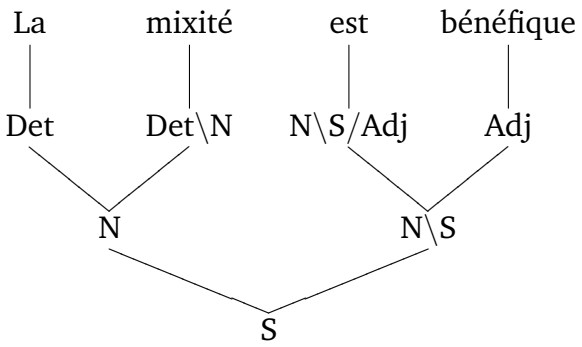


FIGURE 4.3 – Arbre de dérivation pour la phrase « La mixité est bénéfique ».

Dans les grammaires catégorielles de dépendances, les types primitifs (l'ensemble Pr) représentent des fonctions syntaxiques (les noms des dépendances employées dans les représentations en dépendances e.g. *pred*, *obj*) et où S, l'axiome, correspond au nom de la dépendance racine d'une phrase. Nous appelons également sous-types les types primitifs apparaissant dans les types appartenant à l'ensemble Tp. Alors, dans le cas projectif, à tout mot α du lexique est associé un sous-ensemble de types de la forme $B\backslash A/C$ (B et C peuvent éventuellement être vides) où :

- A est un type primitif représentant le nom de la dépendance entrante (i.e. arrivant sur le mot α dans la structure de dépendances), également désigné comme le **sous-type tête** ;
- $B = B_1\backslash ...\backslash B_n \in Tp$ tel que $\forall i \in \{1,...,n\}$, B_i est un type primitif correspondant au nom d'une dépendance sortante à gauche ;
- $C = C_1/.../C_m \in Tp$ tel que $\forall j \in \{1,...,m\}$, C_j est un type primitif correspondant au nom d'une dépendance sortante à droite.

La sous-structure de dépendances engendrée à partir de α, conformément au type de la forme $B\backslash A/C$ qui lui est associé, est présentée par la figure 4.4.

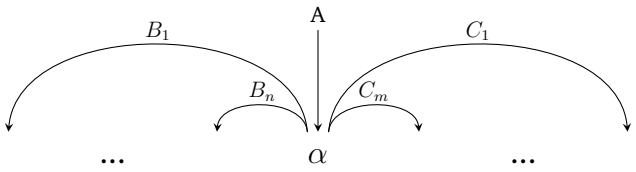


FIGURE 4.4 – Dépendance projective entrante et dépendances projectives sortantes pour un mot α associé à un type de la forme $B_1\backslash ...\backslash B_n\backslash A/C_1/.../C_m$.

En outre, les sous-types correspondent, dans une structure de dépendances, à des dépendances projectives (donc également aux ancres) tandis que les dépendances non-projectives sont gérées par des valences polarisées. Les valences polarisées sont représentées par des types primitifs combinés à des polarités. Elles sont de quatre sortes, avec A un type primitif :

- une valence négative gauche (ou sud-ouest) de la forme $\swarrow A$ correspond à une dépendance non-projective gauche entrante ;
- une valence positive gauche (ou nord-ouest) de la forme $\nwarrow A$ correspond à une dépendance non-projective gauche sortante ;

- une valence négative droite (ou sud-est) de la forme $\searrow A$ correspond à une dépendance non-projective droite entrante ;
- une valence positive droite (ou nord-est) de la forme $\nearrow A$ correspond à une dépendance non-projective droite sortante.

Un couple formé par les valences $\swarrow A$ et $\nwarrow A$ représente une dépendance non-projective gauche étiquetée A et un couple formé par les valences $\nearrow A$ et $\searrow A$ représente une dépendance non-projective droite étiquetée A. Les valences $\swarrow A$ et $\nwarrow A$ ou $\nearrow A$ et $\searrow A$ sont également désignées par le terme valences duales. Dans la description des types associés à des mots d'où arrive ou part une dépendance non-projective, cela se traduit par l'ajout des valences en exposant. C'est à dire que pour un mot recevant une dépendance non-projective étiquetée A, le type qui lui est associé est de la forme $[B]^{\swarrow A}$ ou $[B]^{\searrow A}$ (respectivement pour une dépendance gauche ou droite) avec $B \in Tp$. De même, le type associé à un mot d'où sort une dépendance non-projective étiquetée A est de la forme $[B]^{\nwarrow A}$ ou $[B]^{\nearrow A}$ avec $B \in Tp$.

Par ailleurs, chaque dépendance non-projective étant couplée à une ancre, les types des mots recevant une dépendance non-projective comprennent un sous-type tête particulier. Ainsi, un mot recevant une dépendance non-projective étiquetée A est associé à un type de la forme $[B]^{\swarrow A}$ où $B = \ldots \backslash \#\swarrow A / \ldots$ car $\#\swarrow A$ correspond au nom d'une dépendance projective entrante : une ancre. La figure 4.5 présente les sous-structures de dépendances générées à partir de mots recevant une dépendance non-projective.

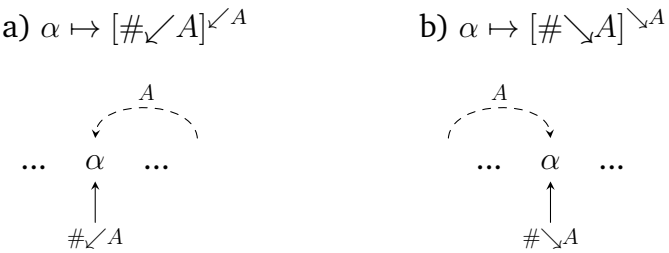


FIGURE 4.5 – Dépendances non-projectives (en pointillé) et ancres (au dessous) entrantes pour un mot α recevant une dépendance non-projective gauche (a) et droite (b). La polarité apparaissant dans le nom de l'ancre ne définit pas sa direction mais la direction depuis laquelle la dépendance non-projective qui lui est associée arrive. Une ancre peut venir de droite ou de gauche indépendamment de la direction de la dépendance non-projective à laquelle elle est couplée.

Pour illustrer l'emploi des types dans la grammaire catégorielle de dépendances, prenons en exemple la phrase dont la structure de dépendances est donnée par la figure 4.6. Considérant une grammaire catégorielle du français, on a, pour chaque mot de la phrase, les types suivants :

Personne	$\mapsto$	$[neg]^{\nearrow compos\text{-}neg}$
par	$\mapsto$	$[circ/prepos\text{-}l]$
la	$\mapsto$	$[det]$
guerre	$\mapsto$	$[det\backslash prepos\text{-}l]$
ne	$\mapsto$	$[\#\searrow compos\text{-}neg]^{\searrow compos\text{-}neg}$
devient	$\mapsto$	$[\#\searrow compos\text{-}neg\backslash circ\backslash neg\backslash S/fs/a\text{-}copul]$
grand	$\mapsto$	$[a\text{-}copul]$
.	$\mapsto$	$[fs]$

Les valences polarisées sont portées par les mots « Personne » et « ne » impliqués dans une relation non-projective de type *compos-neg* i.e. une négation.

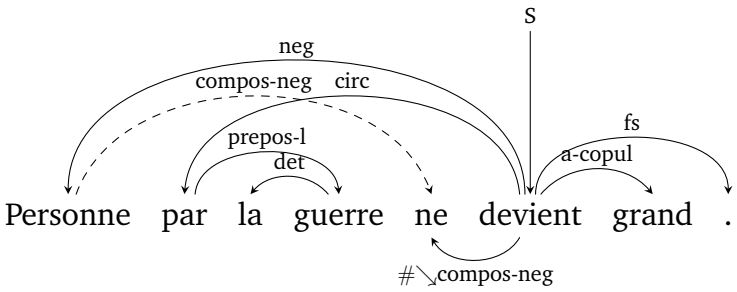


FIGURE 4.6 – Structure de dépendances pour la phrase « Personne par la guerre ne devient grand. ».

L'ajout de la notion de valence polarisée dans les grammaires catégorielles de dépendances permet de gérer les dépendances non-projectives, et ainsi de rendre plus expressives ces grammaires par rapport aux grammaires catégorielles classiques qui, comme nous l'avons vu, ne seraient capables de gérer que les dépendances projectives dans le cadre de l'application de ces grammaires à l'analyse en dépendances. Avec les CDG, il est possible de décrire certains langages contextuels tel que le langage $\{a^n b^n c^n \mid n > 0\}$.

4.3.3 Règles et calculs

Les grammaires catégorielles de dépendances emploient des règles de dérivation particulières permettant de réduire les types des dépendances projectives de manière classique (conformément aux grammaires catégorielles classiques) et de réduire les valences polarisées à l'aide de l'introduction d'une nouvelle règle. Le tableau 4.1 présente les quatre règles de réduction gauches des grammaires catégorielles de dépendances. Quatre règles de réduction droites symétriques existent. Ces règles peuvent être employées dans un système de déduction (où les types des mots d'une phrase forment l'ensemble des hypothèses Γ) ou dans un arbre de dérivation pour prouver qu'une phrase appartient au langage engendré par une grammaire catégorielle de dépendance i.e. $\Gamma \vdash S$.

$\mathbf{L}^l$	$C^{P_1}\,[C\backslash\beta]^{P_2} \vdash [\beta]^{P_1 P_2}$
$\mathbf{I}^l$	$C^{P_1}\,[C^*\backslash\beta]^{P_2} \vdash [C^*\backslash\beta]^{P_1 P_2}$
Ω^l	$[C^*\backslash\beta]^{P} \vdash [\beta]^{P}$
$\mathbf{D}^l$	$\alpha^{P_1(\swarrow C)P(\nwarrow C)P_2} \vdash \alpha^{P_1 P P_2}$, si $(\swarrow C)(\nwarrow C)$ satisfait le principe **FA** dans le potentiel

Principe **FA** (*First Available*) : les valences duales les plus proches forment un couple.

TABLE 4.1 – Règles gauches des grammaires catégorielles de dépendances. Des règles symétriques, désignées par les symboles $\mathbf{L}^r$, $\mathbf{I}^r$, Ω^r et $\mathbf{D}^r$, sont utilisées dans le cas des dérivations à droite. P,P_1 et P_2 sont des potentiels, $C \in Pr$ et α,$\beta \in Tp$.

La règle **L** permet d'éliminer des sous-types par réduction, comme pour les grammaires catégorielles classiques. De plus, à chaque type peut être attaché un potentiel i.e. une liste de valences polarisées pouvant être vide. En ce cas, les potentiels des types (hypothèses) impliqués dans la réduction, P_1 et P_2, sont concaténés pour former un nouveau potentiel $P_1 P_2$. En ce qui concerne la règle **I**, les potentiels sont traités de la même manière tandis que la réduction s'effectue sur les sous-types itérables[3]. La règle Ω permet d'éliminer un type itérable en conservant le potentiel tel qu'il était. Enfin, l'élimination des valences bien appariées (règle **D**) dans le potentiel du type

[3]Les sous-types itérables sont des sous-types que l'on peut réduire plusieurs fois (infiniment) ou pas du tout. Dans

dérivé se fait sur le principe **FA** (First Available). Des valences duales sont des valences de même catégorie dont les polarités sont $\swarrow$ et $\nwarrow$, ou $\nearrow$ et $\searrow$. Le principe FA indique que les valences duales les plus proches dans un potentiel forment un couple c'est à dire que dans un potentiel $P_1(\swarrow C)P(\nwarrow C)P_2$, $(\swarrow C)$ et $(\nwarrow C)$ (valences duales) n'apparaissent pas dans P.

Un exemple de déduction pour la phrase présentée dans la figure 4.6 est donné dans la figure 4.7. Dans cet exemple, les hypothèses correspondent aux types associés aux mots exposés dans la sous-section précédente. On y observe l'application de la règle **L** à droite et à gauche, ainsi que la concaténation des potentiels et l'élimination du couple de valences duales $\nearrow$*c-n*$\searrow$*c-n* correspondant à une dépendances non-projective droite de type *compos-neg*.

$$\dfrac{[neg]^{\nearrow c\text{-}n} \qquad \dfrac{\dfrac{[circ/prep] \qquad \dfrac{[det] \quad [det\backslash prep]}{[prep]}\,\mathbf{L}^l}{[circ]}\,\mathbf{L}^r \qquad \dfrac{[\#_{\searrow c\text{-}n}]^{\searrow c\text{-}n} \qquad \dfrac{\dfrac{[\#_{\searrow c\text{-}n}\backslash circ\backslash neg\backslash S/fs/a\text{-}c] \quad [a\text{-}c]}{[\#_{\searrow c\text{-}n}\backslash circ\backslash neg\backslash S/fs]}\,\mathbf{L}^r \quad [fs]}{[\#_{\searrow c\text{-}n}\backslash circ\backslash neg\backslash S]}\,\mathbf{L}^r}{[circ\backslash neg\backslash S]^{\searrow c\text{-}n}}\,\mathbf{L}^l}{[neg\backslash S]^{\searrow c\text{-}n}}\,\mathbf{L}^l}{[S]^{\nearrow c\text{-}n\searrow c\text{-}n}}\,\mathbf{L}^l}{[S]}\,\mathbf{D}^r$$

FIGURE 4.7 – Preuve d'appartenance de la phrase « Personne par la guerre ne devient grand . » à la grammaire catégorielle de dépendances du français. Les symboles *c-n* et *a-c* correspondent respectivement aux types primitifs *compos-neg* et *a-copul* de la grammaire catégorielle de dépendances du français.

4.3.4 CDG étendues

Nous avons vu dans les sous-sections précédentes des exemples dans lesquels des types sont associés aux mots des phrases. Techniquement, dans les grammaires catégorielles de dépendances, les types ne sont pas associés directement aux mots. En effet, dans le cadre du développement manuel de grammaires, il est plus judicieux d'assigner les types à des ensembles de mots qu'aux mots eux-mêmes, d'une part, dans le but de réduire l'effort manuel et d'autre part car les langues évoluent (fréquemment au niveau du lexique). Ainsi, l'ensemble des symboles terminaux (i.e. les mots) a été réduit à un ensemble de classes grammaticales plutôt qu'à un ensemble de mots et une table de correspondance entre les classes et les mots a été créée. Ainsi, dans les grammaires, à chaque classe grammaticale est associé un ensemble de types, plutôt qu'à chaque mot. Et chaque mot peut appartenir à plusieurs classes grammaticales. Par conséquent, des types associés à différentes classes peuvent être associés à un même mot lors d'une analyse avec les CDG.

Par ailleurs, la complexité de l'algorithme utilisé pour l'analyse en dépendances avec les CDG dépend directement du nombre de types inclus dans ces grammaires. Alors, pour réduire le nombre de types des grammaires tout en conservant la couverture syntaxique de celles-ci il fut nécessaire de regrouper les types similaires à l'aide d'expressions régulières. Ces expressions permettent de définir des ensembles de types et ce sont ces expressions qui sont alors assignées aux classes grammaticales. Dans le cas des CDG, ces expressions sont appelées *regular type expression* (RTE).

---

une structure de dépendances, il s'agit donc de dépendances itérées i.e. de dépendances d'un même type pouvant être rattachées plusieurs fois à un même mot. Par exemple, dans la grammaire du français le sous-type *circ* est souvent itérable car il est possible de pouvoir attacher plusieurs compléments circonstanciels à un même mot, e.g. dans la phrase « Elle était assise dans le parc sous le cerisier. » les syntagmes circonstanciels « dans le parc » et « sous le cerisier » sont attachés à « assise ».

Définissons un ensemble S_{exp} comprenant les sous-expressions pouvant apparaître dans les RTE tel que $\forall i \in \mathbb{N}$, $T_i \in Pr$, alors :

- $T_i \in S_{exp}$;
- $T_i^* \in S_{exp}$ (itération possible du sous-type T_i) ;
- $T_i^? \in S_{exp}$ (présence optionnelle du sous-type T_i) ;
- $(U_1 \mid \ldots \mid U_n) \in S_{exp}$ avec $U_i = T_i \mid T_i^* \mid T_i^?$;
- $(T_1 \mid \ldots \mid T_n)^? \in S_{exp}$;
- $(T_1 \mid \ldots \mid T_n)^* \in S_{exp}$;

Alors les RTE s'écrivent sous la forme :

$$Flr[Fl\backslash B_m\backslash ...\backslash B_1\backslash A/D_1/.../D_l/Fr]^P$$

avec B_j, $D_k \in S_{exp}$ ($\forall j \in \{1,...,m\}$ et $k \in \{1,...,l\}$ pour $k,l \in \mathbb{N}$) des sous-expressions, A de la forme $(T_1 \mid ... \mid T_q)$ pour $q \in \mathbb{N}$ et avec Fl, Fr et Flr de la forme $\{R_1, \ldots, R_n\} \mid \epsilon$ où $\forall i \in \{1,...,n\}$ pour $n \in \mathbb{N}$, $R_i \in S_{exp}$ sont des sous-expressions. Fl, Fr et Flr expriment la flexibilité (respectivement à gauche, à droite et des deux côtés) dans l'ordre des sous-types dans les sous-expressions.

Par exemple, si l'on souhaite définir le contexte syntaxique d'un nom commun en indiquant qu'il peut être un objet (*obj*) ou un sujet (*pred*), qu'il possède un déterminant (*det*) à gauche, qu'il peut avoir un attribut (*attr*) à droite et des modificateurs (*modif*) en nombre indéterminé à droite ou à gauche alors on décrit son type par la RTE suivante : $\{modif^*\}[det\backslash(pred \mid obj)/attr?]$.

4.4 Grammaire du français et corpus

Nous présentons dans cette section la grammaire catégorielle de dépendances du français et les principaux corpus en dépendances du français qui furent développés conjointement et que nous utilisons dans nos travaux.

4.4.1 Grammaire catégorielle de dépendances du français

La grammaire catégorielle de dépendances du français (CDGFr), développée par Dikovsky (2011), est la grammaire catégorielle de dépendances la plus aboutie. Elle comprend à ce jour plus de 3000 RTE permettant une large couverture de la syntaxe du français.

Dans un premier temps, le développement de la CDGFr débuta conjointement avec la création des classes grammaticales associées aux RTE et la construction d'une base de données lexicale permettant de faire correspondre mots et classes grammaticales. Par la suite, l'intégration des informations du Lefff (Sagot, 2010), un lexique des formes fléchies du français comprenant 536 375 entrées associées à 110 477 lemmes, modifia la classification lexicale de la CDGFr. Il en résulte à ce jour un jeu de 185 classes grammaticales incluant des informations morpho-syntaxiques très précises adaptées à la syntaxe du français. Elles peuvent être regroupées en 28 classes plus générales dont 18 permettant de catégoriser les mots et 10 permettant de catégoriser les signes de ponctuation. Les 28 classes grammaticales générales sont présentées dans la table 4.2.

Chaque RTE comprend un sous-type tête (voire un choix entre plusieurs sous-types têtes) correspondant au nom de la dépendance entrante sur un mot, i.e. il s'agit du sous-type A dans les

Adjectifs	**Adj**	Prépositions	**PP**	Ponctuations	**Dash**
Adverbes	**Adv**	Verbes auxiliaires	**Vaux**		**Parentheses**
Collocations	**Colloc**	Verbes copules	**Vcopul**		**QuestMark**
Conjonctions	**Conj**	Verbes intransitifs	**Vi**		**Quotes**
Déterminants	**Det**	Verbes substituts	**Vlight**		**SemiColon**
Interjections	**Expletives**	Verbes transitifs	**Vt**		**Chevrons**
Noms	**N**	Verbes ditransitifs	**V2t**		**Colon**
Nombres	**Num**	Unités inconnues	**UT**		**FullStop**
Partitifs	**Part**				**EmphatMark**
Pronoms	**PN**				**Comma**

TABLE 4.2 – Liste des classes grammaticales générales de la CDGFr

expressions de la forme $B\backslash A/C$. Ces noms/types de dépendances sont au nombre de 116 dans la version de la grammaire que nous employons pour nos travaux et peuvent être regroupés en 38 groupes (voir l'annexe A pour plus de détails). Ils représentent des fonctions syntaxiques précises, parfois spécifiques au français mais dont la majorité sont adaptables à d'autres langues. Les noms des dépendances, ainsi que les groupes auxquels ils appartiennent sont présentés en annexe A. Parmi ces noms de dépendances, 89 sont exclusivement associés à des dépendances projectives et 23 peuvent être associés à des dépendances projectives ou non-projectives. Finalement, seulement 4 noms de dépendances sont exclusivement associés à des dépendances non-projectives. Il s'agit de cas particuliers d'agrégation, de comparaison, de négation et de d'objet copule.

4.4.2 Corpus en dépendances

Chacun des travaux présentés dans cette thèse est relié d'une manière ou d'une autre aux grammaires catégorielles de dépendances et à la représentation qui en découle. Chacun des traitements mis en place au cours de cette thèse est donc testé sur un ensemble de données préalablement annoté selon le schéma d'annotation induit par les grammaires catégorielles de dépendances. Par ailleurs, les différents outils d'apprentissage que nous intégrons dans nos processus exploitent également ces données.

Les données furent annotées, parallèlement au développement de la CDGFr, par Alexandre Dikovsky et Danièle Beauquier. Il s'agit de phrases du français provenant de multiples sources : d'œuvres littéraires du 19e et 20e siècles (« La ronde et autres faits divers » de J.M.G Le Clezio, « L'étranger » de A. Camus, « Voyage au bout de la nuit » de L.F. Céline, « Germinal » de E. Zola), de journaux (article du monde « L'enfance de l'Univers dévoilée ») et de sources diverses[4].

Il en résulte un corpus de 3 030 phrases du français que nous appelons CDG Treebank. Ce corpus, que nous exploitons dans nos travaux, nous a été fourni pas les annotateurs mais n'est néanmoins pas encore disponible librement. Chaque phrase est segmentée en mots qui sont eux-même chacun étiqueté par une classe grammaticale de la CDGFr puis attaché à leur gouverneur par une dépendance étiquetée (en conformité avec les types proposés par la CDGFr). Les statistiques précises du corpus, concernant le nombre de phrases, de mots et les taux de non-projectivité de chacun des sous-corpus, sont exposées dans le tableau 4.3. On constate que le pourcentage de dépendances non-projectives varie selon les textes et que le pourcentage de phrases non-projectives, variant aussi fortement, est important pour chacun des sous-corpus. Globalement, le CDG Treebank est un corpus intéressant pour l'analyse en dépendances du français et en particulier pour

[4]Le corpus *CDG devel* est principalement composé de phrases de la vie courante et de phrases extraites d'articles journalistiques. Elles furent utilisées pour le développement de la CDGFr.

Corpus	genre	nb ph.	% ph. non-proj	nb mots	% dép. non-proj
CDG devel	divers	1 941	43,48	21 598	4,96
Le Clezio	littérature	530	28,87	9 924	1,90
Camus L'étranger	littérature	319	49,53	5 253	4,11
Céline Voyage	littérature	91	39,56	1 801	3,78
Zola Germinal	littérature	85	44,71	2 497	2,08
Universe	journal	64	32,81	1 619	1,30
CDG Treebank		3 030	41,25	42 692	3,79

TABLE 4.3 – Statistiques des corpus en dépendances du CDG Lab - Noms des corpus, nombre de phrases, pourcentage de phrases non-projectives, nombre de mots, pourcentage de dépendances non-projectives. Le corpus CDG Treebank correspond à la réunion des sous-corpus.

les travaux incluant la gestion des dépendances non-projectives nécessaires à la bonne représentation de la syntaxe du français. Parmi les dépendances non-projectives apparaissant dans le CDG Treebank, les plus fréquentes sont les dépendances de type clitique, négation, objet, refléxif et coprédication. Les clitiques et les négations sont associés la plupart du temps à des dépendances non-projectives courtes (pour lesquelles un ou deux mots séparent le gouverneur du subordonné) tandis que les coprédications ou les appositions sont fréquemment associées à des dépendances de longue distance.

4.5 CDG Lab

Le **CDG Lab**, décrit dans Béchet *et al.* (2014), est un environnement intégré proposant de réaliser plusieurs tâches :

- développer et maintenir des grammaires catégorielles de dépendances pour diverses langues ;
- utiliser ces grammaires pour l'analyse en dépendances ;
- construire et modifier des corpus.

La grammaire et le corpus du français présentés dans la section précédente ont été développés à l'aide du CDG Lab. Nous ne discutons pas dans cette section du développement des grammaires catégorielles de dépendances mais seulement de l'analyseur intégré permettant de réaliser des analyses en dépendances et de construire des corpus en dépendances.

Le fonctionnement de l'analyseur du CDG Lab repose sur le formalisme des grammaires catégorielles de dépendances. Pour effectuer une analyse, la première étape est donc la sélection d'une grammaire. À ce jour, la grammaire ayant enregistré l'évolution la plus importante est la grammaire du français. Cependant, sont aussi exploitables, des grammaires pour le russe, l'allemand, le hollandais et l'anglais.

Selon le but recherché, analyse pure ou construction de corpus, il est possible d'exécuter une analyse à partir d'un ensemble de phrases téléchargées ou d'un corpus existant et d'en requérir les scores d'attachement ou à partir de la saisie directe d'une phrase. Dans tous les cas, le mécanisme d'analyse étant fondé sur un algorithme adapté de CYK, l'analyseur peut produire une liste de toutes les analyses possibles pour chaque phrase analysée. La longueur de la liste pouvant être exponentielle suivant la taille de la phrase, aboutir à une solution en un temps convenable n'est

pas garanti. L'analyseur offre alors la possibilité à l'utilisateur de choisir le nombre maximum de structures de dépendances à calculer et le temps d'analyse à ne pas dépasser (dans ce contexte l'analyseur a une complexité polynomiale).

Le fonctionnement global du processus d'analyse comprend, dans un premier temps, le découpage de la phrase en token et le regroupement en mots possibles[5] reconnus par le lexique (augmenté d'expressions régulières pour la reconnaissance des chiffres par exemple). Puis, l'analyse passe par le remplissage d'une matrice triangulaire, dont la première ligne, comprenant une case par mot, est complétée par les types (RTE) possibles[6] de chacun de ces mots. La matrice est ensuite remplie à l'aide de l'algorithme adapté de CYK, appliquant les règles de réduction des CDG (voir la sous-section 4.3.3) pour les dépendances projectives et les ancres. La combinaison des valences polarisées, réalisant l'ajout des dépendances non-projectives, est calculée dans un second temps. La construction des structures de dépendances est réalisée dans une dernière étape.

La description précédente fait état d'un mode d'analyse autonome dans lequel toutes les possibilités d'analyses sont prises en compte. L'analyse peut ne pas aboutir par manque de temps ou d'espace ou bien l'analyseur peut proposer en premier des analyses qui ne sont pas les meilleures. Pour pallier à ces problèmes, deux autres modes d'analyses sont proposés à l'utilisateur :

- le mode d'analyse par sélection des têtes. Il permet de réduire l'espace de recherche ;
- le mode d'analyse par approximation. Il permet de trouver la bonne analyse dans l'ensemble des solutions potentielles.

Il s'agit de modes d'analyse que l'on pourrait qualifier de pré et post processus d'annotation manuelle. Ils sont donc principalement employés pour la construction de corpus. La finalité commune de ces deux modes d'analyse est de restreindre l'espace de recherche de l'analyseur et d'obtenir la bonne structure de dépendances en première position parmi les sorties de l'analyseur.

Le mode d'analyse par sélection des têtes consiste à inclure l'intervention d'un utilisateur dans le choix du découpage de la phrase et de ses annotations. Après avoir découpé et regroupé les tokens, l'analyseur à travers un tableau de sélection des têtes (e.g. figure 4.8) propose à l'utilisateur de :

- confirmer ou infirmer les regroupements de tokens ;
- sélectionner ou non la classe grammaticale de chaque mot ;
- sélectionner ou non le sous-type tête (ou le groupe) de chaque mot.

Ces choix permettent alors de restreindre considérablement le nombre de types possibles par mot et ainsi le nombre d'analyses possibles, dans le but d'obtenir une bonne solution rapidement.

La figure 4.8 illustre l'étape de sélection des têtes (comprenant le choix des mots et des classes grammaticales). Dans ce tableau chaque découpage possible en mots est proposé (e.g. l'expression « à juste titre » est d'une part proposée telle quelle et d'autre part découpée en trois mots simples). Pour chaque mot possible, toutes les classes grammaticales (générales) et tous les groupes syntaxiques qui peuvent lui être associés sont proposés (voir la sous-section 4.4.1 pour plus de détails sur les classes et les groupes). Pour chaque classe grammaticale générale proposée il est possible de voir la liste des classes grammaticales complètes de même catégorie et de sélectionner la bonne. Pour chaque groupe syntaxique proposé il est également possible de voir la liste des sous-types têtes (noms de dépendances) qui appartiennent à ce groupe et de sélectionner la tête correcte.

[5] Toutes les combinaisons possibles entre des tokens consécutifs sont conservées donc il s'agit potentiellement de mot composés ou d'expressions multi-mots.

[6] À partir des classes grammaticales possibles de chaque mot est déduit la liste des RTE possibles pour chaque mot.

Det Det(Lex=art|pn) N	N	Adj N Vaux Vcopul	Adj V2t Vcopul Vt	Conj PP	Adv Conj	Adj Adv Colloc Conj N	N Vt	FullStop
All classes	All classes	All classes	All classes	All classes	All classes	All classes	All classes	All classes
None	None	None	None	None	None	None	None	None
Son	travail	est	reconnu	à	à juste titre	juste	titre	.
All types	All types	All types	All types	All types	All types	All types	All types	All types
AGGR APPOS CIRC CLAUS CONJ COORDV COPRED COPUL CORREL DET JUNC OBJ PRED PREPOS RESTRICT SENT VOCATIVE	AGGR APPOS CIRC CLAUS CONJ COORDV COPRED COPUL CORREL JUNC OBJ PRED pred PREPOS RESTRICT SENT VOCATIVE	AGGR APPOS CIRC CLAUS CONJ COORDV COPRED COPUL CORREL EMPHAT EXPLET JUNC MODIF OBJ PRED PREPOS RESTRICT SENT VOCATIVE	AGGR APPOS AUX CIRC CLAUS CONJ COORDV COPRED COPUL EMPHAT EXPLET JUNC MODIF OBJ PRED PREPOS SENT VOCATIVE	AGGR APPOS ATTR CIRC CLAUS CONJ COORDV COPRED COPUL CORREL EMPHAT INF JUNC OBJ PRED REL SENT	APPOS ATTR CIRC CONJ COPUL EMPHAT EXPLET JUNC PREPOS prepos-g prepos-l prepos-o SENT	AGGR APPOS ATTR CIRC CLAUS CONJ COORDV COPRED COPUL CORREL EMPHAT EXPLET JUNC MODIF OBJ PRED PREPOS RESTRICT SENT VOCATIVE	AGGR APPOS CIRC CLAUS CONJ COORDV COPRED COPUL CORREL EXPLET JUNC OBJ PRED PREPOS RESTRICT SENT VOCATIVE	JUNC PUNCT

FIGURE 4.8 – Tableau de sélection des têtes pour la phrase « Son travail est reconnu à juste titre. ».

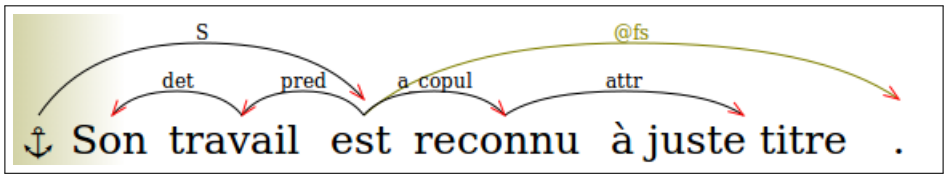


FIGURE 4.9 – Structure de dépendances incorrecte en sortie de l'analyseur du CDG Lab pour la phrase « Son travail est reconnu à juste titre. ».

Le mode d'analyse par approximation intervient après l'achèvement d'une première analyse. L'analyseur propose une ou plusieurs structures de dépendances (e.g. figure 4.9) et l'utilisateur peut alors annoter positivement ou négativement chaque dépendance de la ou des structures (e.g. figure 4.10). Les annotations permettent à l'analyseur de définir des poids sur les dépendances et ainsi d'influer sur le tri des structures de dépendances, la meilleure structure de dépendances étant celle ayant le plus d'annotations positives et le moins d'annotations négatives par rapport aux choix de l'utilisateur. En détail, l'analyseur sélectionne d'abord les analyses ayant le moins de dépendances négatives puis en cas d'égalité celles qui ont le plus de dépendances positives. L'analyse par approximation est applicable autant de fois que nécessaire jusqu'à l'obtention d'une structure de dépendances correcte (e.g. figure 4.11).

L'analyseur du CDG Lab permet ainsi l'annotation de structures de dépendances dans le but de construire des corpus en dépendances dont les structures sont conformes à la syntaxe décrite par les grammaires catégorielles de dépendances.

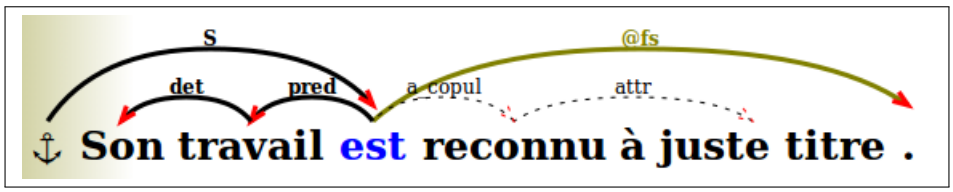


FIGURE 4.10 – Structure de dépendances manuellement annotée pour la phrase « Son travail est reconnu à juste titre. ». Les dépendances en gras sont annotées positivement : l'étiquette et la dépendance sont correctes. Les dépendances en pointillés sont annotées négativement : l'étiquette et/ou la dépendance est incorrecte.

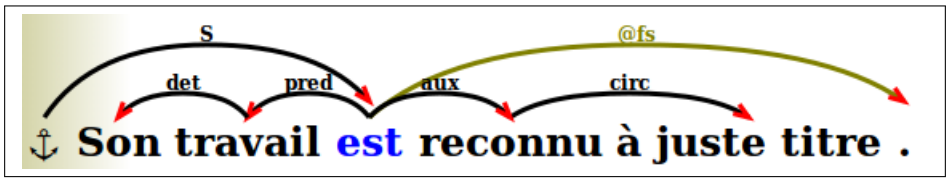


FIGURE 4.11 – Structure de dépendances correcte pour la phrase « Son travail est reconnu à juste titre. » annotée positivement par un annotateur.

4.6 Conclusion

Les grammaires catégorielles de dépendances ont permis d'engendrer une représentation en dépendances mixte projective/non-projective. Le couplage des dépendances non-projectives avec des dépendances projectives, les ancres, décrit la syntaxe locale et les relations distantes de ces dépendances. Les grammaires catégorielles de dépendances incluent donc la gestion des dépendances non-projectives, enjeu essentiel dans la description de nombreuses langues.

Dès lors, l'évolution de ces grammaires et leur application pratique a permis de mettre en place un outil d'analyse et d'annotation en dépendances conforme aux CDG. En particulier, le développement d'une grammaire du français et de plusieurs corpus en dépendances du français permet aujourd'hui d'étudier plus précisément les cas de non-projectivité dans le français.

Nos travaux se placent dans un contexte d'amélioration des techniques d'analyse en dépendances non-projective avec les grammaires catégorielles de dépendances d'une part et dans l'esprit des grammaires catégorielles d'autre part.

II

Étiquetage syntaxique pour les grammaires catégorielles de dépendances

5

Pré-annotation automatique dans le cadre de l'annotation en dépendances avec les CDG

5.1 Introduction

Aujourd'hui l'analyse en dépendances est de plus en plus étudiée et exploitée dans le domaine du traitement automatique des langues à travers des applications telles que la traduction automatique ou l'extraction d'informations. Certaines applications, comme par exemple les systèmes de question-réponse, ont besoin également d'identifier les dépendances non-projectives dans les structures pour répondre à des problèmes particuliers. L'analyse en dépendances peut donc être vue comme une étape préliminaire à une autre tâche. Dès lors, il est nécessaire que l'étape d'analyse soit effectuée efficacement en terme de rapidité et de précision. Les systèmes dirigés par les données dédiés à l'analyse en dépendances sont particulièrement efficaces en ce sens et donc fréquemment employés, pour l'analyse en dépendances projective ou non-projective. Cependant ces systèmes requièrent un entraînement sur des données annotées suffisantes en terme de quantité et de qualité pour produire des analyses précises et couvrant une large partie des phénomènes syntaxiques des langues.

Dans le cas du français, les principaux corpus syntaxiques disponibles sont des corpus qui furent annotés en constituants d'abord puis convertis en dépendances. Il en résulte des corpus en dépendances majoritairement projectifs (voir la section 2.4 du chapitre 2). Par conséquent, dans le cadre de l'analyse en dépendances supervisée, les analyseurs en dépendances entraînés sur ces données ne sont pas capables de reconnaître les cas de discontinuité de la langue quand bien même les systèmes en sont capables. Le développement de corpus en dépendances contenant des dépendances non-projectives devient alors nécessaire pour utiliser pleinement le potentiel de ces systèmes prenant en compte les cas non-projectifs.

L'environnement CDG Lab (voir la section 4.5 du chapitre 4) propose le développement conjoint d'une grammaire du français (CDGFr) et d'un corpus en dépendances du français (CDG Treebank) contenant des dépendances non-projectives. Le processus d'annotation inclus dans le CDG Lab a

permis la construction d'un premier ensemble de corpus comprenant en tout 3 030 phrases correspondant à 42 692 mots[1] dont 3,8% sont associées à des dépendances non-projectives. Cependant, le processus d'annotation requière deux étapes de pré-annotation manuelle qui alourdissent le travail des annotateurs. En effet, le temps d'analyse d'une phrase de taille relativement longue pouvant être trop important, les annotateurs emploient fréquemment le mode d'analyse par sélection des têtes pour réduire les possibilités d'analyse et ainsi obtenir une première solution plus rapidement. Malheureusement, la sélection des têtes, consistant à remplir manuellement le formulaire de sélection des têtes (i.e. sélection des mots et de leurs étiquettes grammaticales et syntaxiques), est une étape fastidieuse pouvant elle-même prendre du temps aux annotateurs.

À travers nos travaux nous proposons de remplacer l'étape manuelle de sélection des têtes par une étape automatique. Nous souhaitons mettre en place un processus autonome de pré-annotation pour l'analyseur du CDG Lab comprenant la segmentation des phrases en mots et l'étiquetage grammatical et syntaxique de ces mots dans le but d'alléger et d'accélérer le travail des annotateurs.

Nous montrons dans la première section en quoi la sélection des têtes est bénéfique et quel est l'impact de cette sélection des têtes automatique (i.e. sélection des étiquettes de dépendances arrivant sur les mots) sur les résultats de l'analyse en dépendances dirigée par la grammaire du français à travers une première méthode d'étiquetage syntaxique que nous proposons. Le principe de la méthode est de prédire, par une méthode d'étiquetage, les étiquettes de dépendances des mots (i.e. les étiquettes des dépendances devant arriver sur les mots), que nous appelons étiquettes syntaxiques, dans le but de réduire l'ambiguïté de l'étape d'analyse en dépendances.

Notre objectif final étant de s'affranchir de l'intervention des annotateurs avant l'analyse, nous expérimentons, dans la deuxième section, une méthode de sélection jointe des classes grammaticales et des étiquettes de dépendances. Cette première méthode vise à réduire les possibilités d'étiquettes grammaticales et syntaxiques des mots tout en tenant compte de toutes les segmentations possibles des phrases données en entrée de l'analyseur dans le but de mettre en place un premier processus d'analyse en dépendances autonome efficace en terme de rapidité et de précision.

Nous proposons, dans la troisième section, une deuxième méthode consistant en un processus complet de pré-annotation des phrases comprenant la segmentation, l'étiquetage grammatical et la prédiction des étiquettes de dépendances. Nous en présentons les bénéfices pour les annotateurs dans le cadre du processus d'annotation en dépendances complet ainsi que les problèmes engendrés par l'intégration des étapes de pré-annotation sur l'étape de validation manuelle des structures de dépendances par les annotateurs.

5.2 Réduction de l'ambiguïté d'analyse

L'analyseur du CDG Lab est un analyseur dirigé par les règles des grammaires catégorielles de dépendances. Ce type d'analyses (basées sur les grammaires) génère, pour une phrase donnée, toutes les analyses possibles, i.e. toutes les structures de dépendances syntaxiquement correctes pour cette phrase. Sans autres informations que la phrase elle-même, l'analyseur considère, lors d'une analyse, tous les mots, toutes les classes grammaticales possibles pour chacun de ces mots et toutes les possibilités de dépendances entre les mots. Pour des phrases relativement longues, l'espace de recherche peut être très grand et induire des temps d'analyse exponentiels.

[1] Nous rappelons que nous utilisons le terme « mot » dans un sens très général, pour désigner tout segment de texte des phrases du corpus, il s'agit donc de toutes les formes fléchies possibles pour des unités lexicales du français dont éventuellement des mots composés, des expressions multi-mots ainsi que les chiffres/nombres et les signes de ponctuations.

Ces temps d'analyse élevés sont contraignants dans le cadre du processus d'annotation en dépendances d'un ensemble de phrases brutes. Pour un annotateur, l'annotation d'une phrase en dépendances consiste à valider/corriger une structure de dépendances produites à partir d'une étape antérieure d'analyse en dépendances. Or, fréquemment, l'espace de recherche est trop important et induit des temps d'analyse trop élevés ou fait échouer l'analyse. En ce cas, l'annotateur utilise le formulaire de sélection des têtes du CDG Lab pour fournir des informations sur la phrase à l'analyseur et ainsi réduire l'espace de recherche. Le formulaire de sélection des têtes comprend les listes des classes grammaticales et des étiquettes syntaxiques possibles pour chaque mot possible de la phrase à analyser.

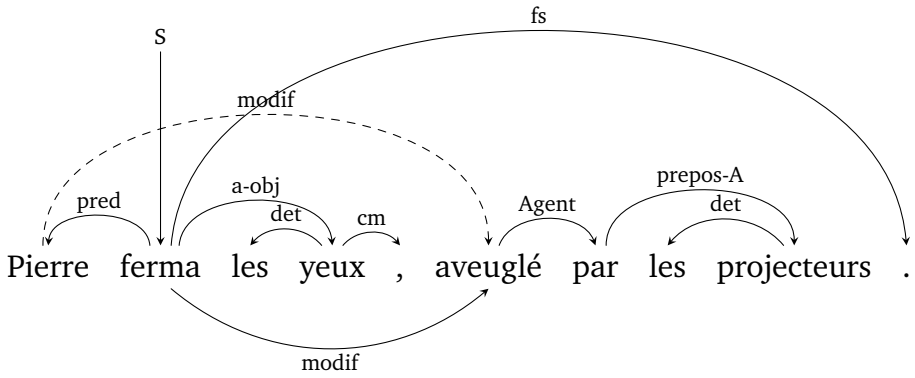


FIGURE 5.1 – Structure de dépendances pour la phrase « Pierre ferma les yeux, aveuglé par les projecteurs. ».

La sélection des têtes consiste à sélectionner pour chaque mot possible d'une phrase son étiquette syntaxique (et éventuellement sa classe grammaticale). L'étiquette syntaxique choisie pour un mot correspond à l'étiquette de la dépendance qui pointera sur le mot dans la structure de dépendances[2]. Par exemple, les étiquettes syntaxiques associées aux mots de la phrase présentée dans la figure 5.1 sont exposées dans la figure 5.2.

Pierre	**ferma**	**les**	**yeux**	**,**	**aveuglé**	**par**	**les**	**projecteurs**	**.**
pred	S	det	a-obj	cm	modif	Agent	det	prepos-A	fs

FIGURE 5.2 – Les étiquettes syntaxiques associées aux mots de la phrase « Pierre ferma les yeux, aveuglé par les projecteurs. ».

Le choix d'une étiquette syntaxique pour chaque mot d'une phrase donnée permet de réduire drastiquement le nombre de types associés à chaque mot et ainsi le nombre de combinaisons possibles à calculer par l'algorithme d'analyse. En effet, les étiquettes syntaxiques correspondent également aux types primitifs des grammaires catégorielles de dépendances (voir la section 4.3.2 du chapitre 4). Une analyse avec sélection des têtes consiste donc à considérer uniquement les types de la grammaire dont le type tête (type primitif central des types définis par les grammaires catégorielles de dépendances i.e. le type primitif A dans un type de la forme $[B\backslash A/C]^P$) correspond aux étiquettes syntaxiques sélectionnées pour chaque mot. Sélectionner une tête pour un mot permet donc de sélectionner un ensemble restreint de types lors de l'analyse. Moins de possibilités

[2]Un mot recevant une dépendance non-projective reçoit également une ancre de même rôle syntaxique (i.e. ayant la même étiquette de dépendance). L'étiquette syntaxique d'un tel mot correspond donc autant à l'étiquette de la dépendance non-projective qui pointe sur ce mot qu'à l'étiquette de l'ancre qui pointe sur ce même mot.

d'analyse existent, ainsi moins de structures de dépendances sont générées ce qui réduit les temps d'analyse.

Par exemple, pour la phrase donnée en exemple dans la figure 5.1, le nombre de structures de dépendances générées par une analyse autonome (sans restrictions sur les types ou les classes) atteint 1 518 tandis qu'une analyse avec pré-sélection des têtes correctes pour chaque mot réduit ce nombre à seulement 2 structures.

Nous proposons, dans cette section, une première méthode automatique de sélection des têtes dans le but d'évaluer son impact sur l'étape d'analyse en dépendances du CDG Lab. La sélection automatique des têtes consiste en une étape d'étiquetage syntaxique où les étiquettes prédites sont les étiquettes des dépendances arrivant sur les mots.

5.2.1 Étiquetage syntaxique

Nous proposons d'effectuer une étape d'étiquetage syntaxique en amont de l'étape d'analyse en dépendances du CDG Lab. Le principe d'une telle tâche est de prédire les étiquettes syntaxiques des mots d'une phrase donnée. Pour cette tâche, la bonne segmentation des phrases en mots et leurs étiquettes grammaticales sont connues. Il s'agit d'un traitement similaire à celui de l'étiquetage grammatical mais dans l'esprit d'une tâche telle que le supertagging (voir la sous-section 3.3.2 du chapitre 3) permettant de réduire l'ambiguïté à l'entrée de l'analyseur. Cependant, dans notre cas, il ne s'agit pas de trouver un ensemble de structures complexes (i.e. des arbres élémentaires dans le cas du supertagging pour les grammaires TAG) conformes à chaque mot mais de prédire une étiquette syntaxique simple correspondant à l'étiquette de la dépendance arrivant sur chaque mot.

Globalement, l'étape d'étiquetage consiste à prédire les étiquettes des dépendances sans prédire les dépendances. Un des enjeux est donc d'étudier la pertinence d'une telle méthode, à savoir, si une méthode d'étiquetage locale permet de prédire correctement les étiquettes des dépendances pour toutes sortes de dépendances (courtes ou longues, projectives ou non-projectives).

En outre, nous disposons d'un ensemble de phrases annotées en dépendances selon le schéma d'annotation induit par les CDG. Pour effectuer l'étiquetage syntaxique des phrases nous employons donc une méthode d'étiquetage supervisée classique adaptée à un jeu d'étiquettes particulier. Ce jeu d'étiquettes syntaxiques correspond à l'ensemble des étiquettes de dépendances définies par la grammaire catégorielle de dépendances du français (les types primitifs).

• **Les données** que nous utilisons pour l'entraînement d'un étiqueteur sont un sous-ensemble des phrases du CDG Treebank (voir la sous-section 4.4.2 du chapitre 4). Ces phrases sont segmentées et étiquetées avec les jeux d'étiquettes grammaticales et syntaxiques de la grammaire de dépendance du français (CDGFr). Il s'agit respectivement de jeux de 185 et 117 étiquettes. Le nombre de classes grammaticales étant important, nous décidons d'une part de sous-catégoriser ces classes pour arriver à deux formes de sous-classification : les classes grammaticales générales (28 classes) et les classes grammaticales étendues (86 classes). Les classes étendues correspondent à des classes générales auxquelles ont été ajoutées des informations grammaticales (voir l'annexe B). D'autre part, les étiquettes syntaxiques de la grammaire sont déjà comprises dans des groupes de dépendances. Pour nos expérimentations, nous utilisons donc deux formes de classification des étiquettes syntaxiques : les étiquettes des dépendances (117) et les étiquettes des groupes de dépendances (39) (voir l'annexe A). Un exemple d'annotation d'une phrase du français, selon les différents niveaux d'étiquetage et les différentes classifications est donnée par la figure 5.3.

Les classes grammaticales générales apportent des informations utiles pour la prédiction des étiquettes syntaxiques. Les classes grammaticales étendues, plus précises, permettent de mieux cibler

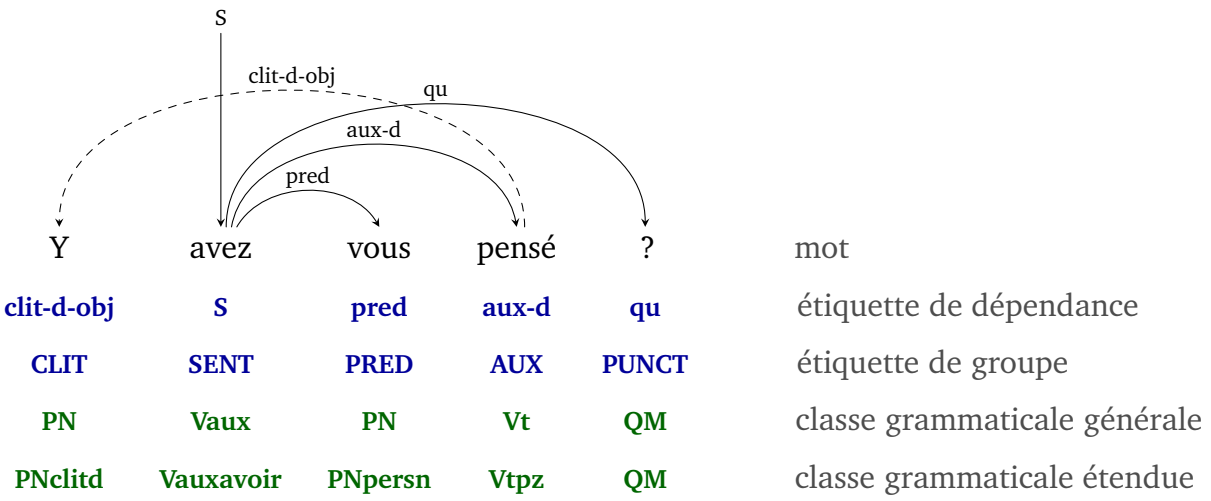


FIGURE 5.3 – Structure de dépendances et annotations grammaticales et syntaxiques pour la phrase « Y avez-vous pensé ? ».

les étiquettes syntaxiques (dans le cas des noms de dépendances comme dans le cas des groupes). La table 5.1 expose le nombre moyen de noms de dépendances ou de groupes possibles pour un mot suivant sa classe grammaticale (générale ou étendue). L'emploi des classes grammaticales étendues comme trait réduit l'ambiguïté sur la prédiction des étiquettes syntaxiques.

		Nombre moyen de			
		noms de dépendances	(max.)	**groupes**	(max.)
Par classe	**générale**	13	(43)	7	(18)
grammaticale	**étendue**	6	(31)	4	(16)

TABLE 5.1 – Nombre moyen (et maximum) de noms de dépendances et groupes possibles par classe grammaticale générale ou étendue.

• **La méthode** que nous employons pour effectuer l'étiquetage syntaxique comprend l'usage des CRF (*Conditional Random Fields*) ou champs markoviens conditionnels. Les CRF, appartenant à la famille des modèles graphiques probabilistes, sont couramment employés dans le domaine de l'étiquetage. Ils permettent de prédire une ou plusieurs séquences d'étiquettes pour une phrase donnée. Nous avons choisi le logiciel Wapiti (Lavergne *et al.*, 2010) pour effectuer l'étiquetage à l'aide des CRF car il est capable de travailler avec des jeux d'étiquettes de grande taille.

Les CRF étant par ailleurs capables de prendre en compte un large choix de traits, le logiciel laisse la possibilité de lui fournir un ensemble de patrons de traits permettant d'extraire des informations variées sur les mots et leurs contextes. Nous disposons pour la tâche d'étiquetage de la bonne segmentation des phrases en mots et de leurs étiquettes grammaticales. Pour la définition des patrons de traits, il est donc possible de choisir une largeur de fenêtre (autour d'un mot) pour indiquer si l'on tient compte des mots et des classes grammaticales précédentes et suivantes lors de la prédiction d'une étiquette syntaxique (il n'est pas nécessaire de préciser les étiquettes syntaxiques précédentes dans les patrons de traits puisqu'elles sont prises en compte dans le modèle CRF).

Nous effectuons des tests préliminaires sur une partie du corpus pour sélectionner les traits pertinents pour l'étiquetage syntaxique. Nous choisissons donc de tester des traits classiques dans le domaine de l'étiquetage grammatical (nous retenons également certains traits parmi plusieurs testés tels que l'extraction du suffixe des mots et le fait de savoir si un mot commence par une majuscule) et dans le domaine de l'analyse en dépendances (il est intéressant d'associer mot et classe grammaticale dans un même trait). À travers des tests préliminaires, nous constatons qu'une fenêtre de 5 (2 mots avant, 2 mots après) donne de bons résultats, qu'élargir la fenêtre à 7 pour les mots génère beaucoup de traits pour peu d'améliorations mais qu'élargir la fenêtre à 7 autour des classes grammaticales est beaucoup plus efficace. Les premiers patrons de traits choisis sont présentés dans la table 5.2.

Traits	
sur les mots	w_{-2}, w_{-1}, w, w_1, w_2
sur les classes grammaticales	c_{-3}, c_{-2}, c_{-1}, c, c_1, c_2, c_3
mixtes	w/c
autres	*suffixe de 3 lettres pour* w,
	w_{-1} *commence par une majuscule ?*

TABLE 5.2 – Patrons de traits pour l'étiquetage syntaxique. w correspond au mot courant et c à sa classe grammaticale.

• **Les expérimentations** sont donc effectuées suivant 4 critères, selon la prise en compte des classes grammaticales générales ou étendues dans les traits et selon l'étiquetage réalisé : étiquetage du nom des dépendances ou étiquetage des groupes. Nous procédons à une évaluation croisée sur les données. Le corpus est divisé en 10 parties. Chaque expérimentation comprend une étape d'entraînement exécutée sur 90 % du corpus et une étape d'étiquetage exécutée sur les 10 % restants.

De plus, l'outil Wapiti nous permet d'engendrer les k meilleurs étiquetages pour une séquence donnée. Nous choisissons de produire les 10 meilleures séquences d'étiquettes pour chaque phrase d'entrée. Ces séquences sont potentiellement assez similaires. Souvent, seulement quelques étiquettes varient d'une séquence à une autre. Pour évaluer la qualité de l'étiquetage syntaxique nous calculons le pourcentage de mots pour lesquels la bonne étiquette syntaxique a été prédite parmi les 1, 1 à 2, 1 à 5 ou 1 à 10 étiquettes prédites (que l'on qualifiera de "meilleures étiquettes") pour chaque mot (sachant qu'il n'y a pas toujours 10 étiquettes différentes par mot).

• **Les résultats** de l'évaluation sont présentés dans la table 5.3. La précision de l'étiquetage des noms de dépendances atteint au mieux 91,1 % en considérant uniquement la première meilleure étiquette et 96,6 % en considérant les 10 meilleures (avec l'utilisation des classes grammaticales étendues dans les traits d'entraînement). La précision de l'étiquetage des groupes de dépendances obtient respectivement 91,6 % et 97,1 %. On constate que les résultats sur l'étiquetage des groupes sont meilleurs que ceux sur l'étiquetage des noms de dépendances, l'ambiguïté étant moindre sur les groupes (tableau 5.1). D'autre part, du fait des informations apportées par les classes grammaticales étendues, les résultats des expérimentations effectuées avec les classes grammaticales étendues sont également plus élevés que les résultats des expérimentations avec les classes grammaticales générales (+3,3 pour les noms de dépendances et +1,2 pour les groupes de dépendances au rang 1).

Étiquetage :	noms de dépendances		groupes	
Traits :	classes générales	classes étendues	classes générales	classes étendues
Top 1	87,8	91,1	90,4	91,6
Top 2	90,0	93,2	92,5	93,7
Top 5	92,9	95,5	95,1	96,0
Top 10	94,6	96,6	96,4	97,1

TABLE 5.3 – Évaluation de l'étiquetage syntaxique (noms de dépendances et groupes). Les résultats représentent le pourcentage de mots pour lesquels la bonne étiquette a été prédite parmi les 1, 1 à 2, 1 à 5 ou 1 à 10 meilleures étiquettes.

Néanmoins, comparant les expérimentations effectuées avec les classes grammaticales étendues, on observe que les résultats de l'étiquetage des groupes sont seulement légèrement meilleurs que les résultats sur l'étiquetage des noms de dépendances et donc que la sous-classification étendue est peut-être mieux adaptée à la désambiguïsation des noms de dépendances que des groupes. Cependant, étiqueter les groupes de dépendances plutôt que les noms de dépendances peut avoir une influence importante sur l'étape d'analyse. En effet, assigner un groupe à un mot équivaut à assigner la disjonction des noms de dépendances appartenant à ce groupe. Par exemple, sélectionner le groupe *OBJ* pour un mot équivaut à sélectionner 8 noms de dépendances (voir l'annexe A) pour ce mot.

5.2.2 Analyse en dépendances

Dans cette sous-section, notre but est d'évaluer l'impact de la sélection automatique des étiquettes syntaxiques sur le processus d'analyse en dépendances du CDG Lab en terme de rapidité d'analyse et de précision. Cette étape consiste donc à fournir les informations produites durant l'étape d'étiquetage précédente à l'analyseur en dépendances dans le but de comparer les performances de l'analyseur via une analyse par sélection automatique des têtes à une analyse autonome (à partir des phrases brutes). Dans un premier temps, l'outil d'analyse par sélection des têtes du CDG Lab est adapté à l'assignation automatique d'une ou plusieurs étiquettes syntaxiques.

• **La procédure d'analyse** comprend l'analyse syntaxique en dépendances guidée par les règles de la grammaire catégorielle de dépendances du français et le tri des structures en dépendances produites. L'analyseur du CDG Lab est conçu pour produire l'ensemble des structures de dépendances possibles pour chaque phrase analysée. En outre, les structures de dépendances en sortie de l'analyseur ne sont pas efficacement triées. Or, nous souhaitons avant tout savoir si parmi les structures de dépendances produites pour une phrase donnée se trouve la bonne structure de dépendances (i.e. la structure de dépendances associée à cette phrase dans le corpus annoté en dépendances de référence). Les structures sont alors triées de la plus proche à la plus éloignée de la structure originale. La plus proche étant celle ayant le plus de dépendances en commun[3] avec la structure de référence.

• **Les expérimentations** sont effectuées d'une part sur les données annotées automatiquement durant l'étape d'étiquetage syntaxique avec le mode par sélection des têtes et d'autre part sur les

[3]Une dépendance est commune aux deux structures de dépendances si elle possède dans les deux structures le même gouverneur, le même subordonné et le même nom de dépendance.

phrases brutes avec le mode autonome du CDG Lab. Pour le mode d'analyse par sélection des têtes, nous utilisons les étiquettes syntaxiques ayant obtenues les meilleurs résultats, c'est à dire celles prédites à l'aide des classes grammaticales étendues. L'étape d'analyse en dépendances est donc effectuée d'une part avec la pré-sélection des noms de dépendances et d'autres part avec la pré-sélection des groupes. De plus, chacune des expérimentations est divisée en quatre selon l'attribution de 1, 1 à 2, 1 à 5 ou 1 à 10 étiquettes syntaxiques différentes à chaque mot. Cette attribution correspond aux 10 meilleures séquences d'étiquettes produites durant l'étape d'étiquetage. Le nombre d'étiquettes différentes prédites pour un mot peut donc être inférieur à 10 puisque les séquences d'étiquettes peuvent être similaires.

En outre, parfois, aucune structure de dépendances n'est produite par l'analyseur. Deux raisons sont possibles :

- les noms de dépendances (ou groupes) assignés sont en contradiction avec les règles de la grammaire, cela entraîne alors un échec de l'analyse ;
- le temps d'analyse est trop élevé, l'analyse s'interrompt donc avant d'aboutir.

Dans un premier temps, nous souhaitons comparer le nombre de phrases ayant abouti (i.e. ayant produit au moins une structure de dépendances) pour chaque expérimentation ainsi que le temps d'analyse moyen d'une phrase du corpus. Nous présentons donc exactement pour chaque expérimentation :

- le nombre d'analyses ayant abouti (AA), i.e. ayant produit au moins une structure de dépendances ;
- le nombre d'analyses n'ayant pas abouti par non-conformité des têtes sélectionnées avec la grammaire (NA-C) ou par manque de temps (NA-T), sachant que la limite de temps d'une analyse est une valeur définie au préalable (pour ces expérimentations nous choisissons 10 secondes) ;
- le nombre moyen de mots (UL) par phrase selon les cas (AA, NA-C ou NA-T) ;
- le temps moyen d'analyse d'une phrase ayant abouti (AA) ;

D'autre part, la restriction des étiquettes sur les mots engendre couramment une perte de précision sur l'analyse en dépendances. Notre seconde évaluation consiste alors à mesurer les scores maximums qu'il est possible d'atteindre en restreignant l'espace de recherche par la sélection automatique des têtes. Les scores sont donc calculés pour chaque phrase sur la meilleure structure de dépendances qui lui est associée et uniquement sur les phrases dont l'analyse a abouti (dans ce cadre, les résultats obtenus correspondent plus à des scores de rappel qu'à des scores de précision). Les critères d'évaluation sont :

- le LAS (*Labelled Attachement Score*), le pourcentage de mots pour lesquels la bonne dépendance et la bonne étiquette de dépendance ont été trouvées ;
- le UAS (*Unlabelled Attachement Score*), le pourcentage de mots pour lesquels la bonne dépendance a été trouvée.

Par ailleurs, les résultats sont divisés en deux parties : ils sont calculés dans un premier temps sur tous les mots et dans un deuxième temps uniquement sur les mots recevant une dépendance non-projective dans la structure d'origine.

Analyse autonome

Nb de têtes	Nombre de phrases						mots/phrase			Temps (s/phrase)
	AA	(%)	NA-C	(%)	NA-T	(%)	AA	NA-C	NA-T	AA
-	1 150	(41,4)	3	(00,1)	1625	(58,5)	7,2	7,3	17,6	2,25

Analyse par sélection des noms de dépendances

Nb de têtes	Nombre de phrases						mots/phrase			Temps (s/phrase)
	AA	(%)	NA-C	(%)	NA-T	(%)	AA	NA-C	NA-T	AA
1	1 805	(65,0)	969	(34,9)	**4**	(00,1)	11,5	16,5	52,5	0,10
1 à 2	2 054	(73,9)	718	(25,8)	6	(00,2)	11,6	17,7	56,1	0,12
1 à 5	2 335	(84,1)	438	(15,8)	5	(00,2)	12,0	20,0	49,4	0,15
1 à 10	2 505	(90,2)	262	(09,4)	11	(00,4)	12,2	22,5	42,8	0,19

Analyse par sélection des groupes

Nb de têtes	Nombre de phrases						mots/phrase			Temps (s/phrase)
	AA	(%)	NA-C	(%)	NA-T	(%)	AA	NA-C	NA-T	AA
1	1 931	(69,5)	832	(29,9)	15	(00,5)	11,5	16,9	45,8	0,20
1 à 2	2 172	(78,2)	586	(21,1)	20	(00,7)	11,6	18,6	45,0	0,24
1 à 5	2 439	(87,8)	302	(10,9)	37	(01,3)	11,8	21,6	43,6	0,29
1 à 10	**2 548**	(91,7)	**179**	(06,4)	51	(01,8)	12,0	24,4	41,6	0,39

TABLE 5.4 – Statistiques sur les expérimentations de l'étape d'analyse en dépendances en fonction de la sélection au préalable ou non des têtes (noms de dépendances et groupes).

• **Les résultats** de l'évaluation croisée sur l'étape d'analyse en dépendances sont présentés dans les tableaux 5.4 et 5.5. Dans le premier tableau, on observe, lors de l'utilisation du mode par sélection des têtes (noms de dépendances et groupes), d'une part la réduction du temps d'analyse des analyses abouties (AA) par rapport à une analyse autonome (au mieux -2,15 secondes sur le temps moyen d'analyse d'une phrase) et d'autre part l'augmentation du nombre de phrases qui aboutissent à au moins une analyse (AA) (au mieux 50,3 points sur le pourcentage d'analyses abouties). Le pourcentage de phrases pour lesquelles l'analyse n'aboutit pas à cause de la limite de temps est alors toujours inférieur à 2 % avec à la sélection des têtes. En outre, les phrases qui aboutissent à une analyse ont une longueur moyenne (au mieux) de 12,2 mots avec la sélection des têtes contre une longueur moyenne de 7,2 dans le cas de l'analyse autonome.

En effet, grâce à la sélection des têtes, l'espace de recherche est réduit et en conséquence les temps d'analyses le sont également, ce qui permet d'augmenter considérablement le nombre de phrases pour lesquelles l'analyse aboutie. Les quelques phrases pour lesquelles l'analyse n'aboutit pas par manque de temps sont celles ayant un nombre moyen de mots très élevé (> 41 mots). Néanmoins, dans le cas des analyses avec sélection des têtes, plus d'analyses n'aboutissent pas par non-conformité de la séquence d'étiquettes prédite avec la grammaire. Ce nombre reste élevé lorsque la sélection est restreinte à une tête par mot. Mais lorsque le nombre de têtes sélectionnées par mot augmente, plus d'analyses aboutissent car un choix plus large est proposé à l'analyseur, ce qui engendre moins de conflits avec la grammaire et plus de chances d'obtenir au moins une

analyse syntaxiquement valide. En parallèle, dû à un espace de recherche plus large, le temps d'analyse augmente également.

Notons que la sélection des groupes répond à cette même logique. Sélectionner un groupe plutôt qu'un nom de dépendance laisse un choix plus large d'analyse. Le nombre d'analyses qui aboutissent et le temps d'analyse global sont donc plus élevés dans le cas de la sélection des groupes. Cependant, pour une sélection de 1 à 10 étiquettes la sélection des groupes n'est pas plus avantageuse que la sélection des noms de dépendances. En effet, le nombre d'analyses abouties n'est pas beaucoup plus élevé tandis que le temps d'analyse est doublé. Augmenter le nombre de noms de dépendances sélectionnés par mot serait probablement plus efficace que sélectionner les groupes car la prédiction des noms de dépendances est plus précise que la prédiction des groupes. En outre, l'augmentation de l'espace de recherche due à la sélection des groupes fait que plus de phrases longues dépassent la limite de temps d'analyse et n'aboutissent donc pas.

Analyse autonome[4]

Nb de têtes	Toutes dépendances		Dépendances non-projectives	
	LAS	UAS	LAS	UAS
-	98,3	99,0	92,7	93,2

Analyse par sélection des noms de dépendances

Nb de têtes	Toutes dépendances		Dépendances non-projectives	
	LAS	UAS	LAS	UAS
1	93,7	96,7	92,4	93,7
1 à 2	95,1	97,3	94,3	95,5
1 à 5	96,2	97,8	94,4	95,5
1 à 10	96,4	97,9	94,5	95,4

Analyse par sélection des groupes

Nb de têtes	Toutes dépendances		Dépendances non-projectives	
	LAS	UAS	LAS	UAS
1	93,9	96,7	88,8	93,3
1 à 2	95,1	97,2	90,0	93,7
1 à 5	96,3	97,9	90,5	93,8
1 à 10	96,7	98,0	91,1	94,3

TABLE 5.5 – Scores d'attachement des analyses en dépendances en fonction de la sélection au préalable ou non des têtes (noms de dépendances et groupes).

Les résultats regroupés dans le tableau 5.5 exposent l'évaluation de la précision de l'analyse en dépendances. De manière similaire aux résultats précédents, les résultats des analyses par sélection

[4]Les scores obtenus pour l'analyse autonome n'atteignent pas 100% car il subsiste des incohérences entre les annotations des corpus en dépendances et les types de dépendances utilisés dans la CDGFr, dûes au développement de cette grammaire.

des têtes augmentent avec l'augmentation du nombre de sélections par mot et atteignent globalement des scores très intéressants (96,4 % en LAS avec la sélection des noms de dépendances et 96,7 % en LAS avec la sélection des groupes pour toutes les dépendances et respectivement 94,5 % et 91,1 % pour les dépendances non-projectives). Notons que les résultats globaux des analyses par sélection des groupes ne sont que légèrement plus élevés que les résultats des analyses par sélection des noms de dépendances. L'élargissement de l'espace de recherche par les groupes ne semble pas pertinent pour obtenir une meilleure précision sur les analyses.

Lorsqu'on s'intéresse aux dépendances non-projectives, on remarque que la précision sur ces dépendances est légèrement moins bonne que sur l'ensemble des dépendances (-1,9 avec la sélection des noms de dépendances et -5,6 avec la sélection des groupes). Par ailleurs, la précision des analyses ayant abouti par sélection des groupes est moins bonne dans les cas non-projectifs. Cela peut s'expliquer par le fait que le taux de dépendances non-projectives est moins élevé parmi les dépendances des analyses abouties dans le cas de la sélection des noms de dépendances. Notons qu'il y a de 4,3 % à 4,6 % de dépendances non-projectives parmi les dépendances des analyses abouties avec sélection des noms de dépendances pour 4,8 % à 4,9 % avec la sélection des groupes. Il est donc possible que les phrases comprenant des cas de discontinuités difficiles soient des phrases dont l'analyse n'a pas abouti par la sélection des noms de dépendances mais ait abouti par la sélection des groupes sans pour autant avoir prédit les bons groupes mais permettant un espace de recherche dans lequel au moins une analyse valide existe.

Globalement, la sélection des noms de dépendances permet d'atteindre des scores intéressants pour l'analyse en dépendances, dans le cas projectif comme dans le cas non-projectif. L'étiquetage syntaxique automatique préalable semble approprié à la désambiguïsation de l'analyse en dépendances avec les CDG.

5.2.3 Discussion

Dans cette section, l'objectif était de mettre en évidence le problème dû à l'ambiguïté de l'analyse en dépendances avec les grammaires catégorielles de dépendances et de proposer une première solution afin de mettre en avant les bénéfices qu'apporte l'intégration d'une étape d'étiquetage syntaxique automatique sur l'analyse en dépendances avec les grammaires catégorielles de dépendances.

La méthode d'étiquetage syntaxique que nous proposons est d'une part pertinente vis-à-vis des scores d'étiquetage obtenus et d'autre part vis-à-vis de son intérêt pour l'analyse en dépendances. Bien que les étiquettes syntaxiques soient naturellement liées aux dépendances, la prédiction des étiquettes seules atteint des scores très intéressants. Le modèle statistique utilisé, ici les CRF, exploitant des informations locales (au niveau des mots) est donc bien adapté à la prédiction des étiquettes syntaxiques.

Suite à l'évaluation de l'étiquetage syntaxique des mots, nous incluons cette étape dans le processus d'analyse en dépendances avec les CDG en tant que sélection des têtes automatique. L'objectif attendu quant à la réduction de l'ambiguïté sur l'analyse est pleinement satisfait : les temps d'analyse sont considérablement réduits. Par conséquent, les nombreuses analyses qui n'aboutissaient pas par manque de temps peuvent alors aboutir à un résultat. Cependant, la sélection automatique des têtes n'étant pas toujours syntaxiquement valide, certaines analyses ne parviennent pas à produire un résultat. L'incohérence des étiquettes sélectionnées avec les types de la CDGFr fait qu'aucune structure de dépendances ne peut être calculée. L'élargissement de l'espace de recherche est donc essentiel pour obtenir un taux élevé d'analyses abouties, ce qui peut être fait en acceptant plus d'étiquettes (e.g. les n plus probables) par mot. En conséquence, l'espace de recherche est étendu et le temps d'analyse s'accroît. Le problème revient alors à trouver un compromis entre réduction de l'ambiguïté et réduction du temps d'analyse.

Néanmoins, les cas de non-conformité de l'étiquetage vis-à-vis de la grammaire soulève un autre problème. Dans le cadre de l'analyse en dépendances ou dans le cas du processus d'annotation en dépendances, il n'est pas acceptable que l'analyseur ne fournisse aucune sortie. Dans les cas où aucune structure de dépendances complète ne peut être trouvée conformément aux étiquettes sélectionnées, il serait nécessaire que l'analyseur produise une analyse partielle.

5.3 Pré-sélection jointe des classes grammaticales et étiquettes syntaxiques

L'utilité de la pré-sélection des têtes (étiquettes syntaxiques) dans le cadre d'une analyse avec les CDG n'est plus à démontrer. Nous avons vu précédemment (section 5.2) que la pré-sélection des têtes par une méthode d'étiquetage syntaxique automatique réduisait considérablement le temps d'analyse avec les CDG tout en permettant d'atteindre des scores d'analyse en dépendances intéressants.

Dans le cadre d'un processus d'annotation en dépendances, la pré-sélection automatique des étiquettes syntaxiques permet en outre d'alléger le travail des annotateurs. Cependant, les travaux précédents pré-supposaient la bonne segmentation des phrases en mots et le bon étiquetage grammatical de ces mots. Or les phrases fournies aux annotateurs ne sont ni segmentées ni étiquetées. Cet état de fait revient alors à considérer deux solutions lors de l'intégration d'une étape automatique de pré-sélection des étiquettes syntaxiques dans le processus d'annotation :

- demander aux annotateurs de segmenter et étiqueter les phrases avant analyse ;
- ou prendre en compte la segmentation et l'étiquetage grammatical dans un processus global de pré-annotation automatique.

Faire pré-annoter les phrases par les annotateurs annihile le bénéfice de la pré-sélection automatique des étiquettes syntaxiques qui est d'alléger le travail des annotateurs en leur évitant de remplir le formulaire de sélection des têtes. Nous proposons donc dans cette section une première méthode de « pré-annotation » des phrases prenant en compte les différentes segmentations possibles d'une phrase et leurs différents étiquetages grammaticaux dans le but de dispenser les annotateurs de l'étape de pré-annotation manuelle. L'objectif est de mettre en place un processus permettant d'obtenir directement et rapidement une première analyse pour une phrase donnée que les annotateurs auront seulement à modifier/valider.

La méthode que nous proposons consiste à traiter indépendamment les mots les uns des autres. La difficulté de cette tâche vient du fait que la segmentation n'est pas connue. L'étape de sélection automatique des étiquettes syntaxiques doit donc pouvoir gérer un contexte variable (pour lequel les mots précédents et suivants peuvent être multiples).

5.3.1 Classification et mode de pré-sélection

Sans pré-traitements manuels ou automatiques, les phrases à étiqueter ne sont ni segmentées en mots ni étiquetées grammaticalement. L'étape de sélection des étiquettes syntaxiques doit pouvoir gérer les différentes possibilités de segmentation et d'étiquetage grammatical des mots. Nous souhaitons, à travers ces travaux, prendre en compte toutes ces possibilités pour chaque phrase analysée. Il ne s'agit donc plus ici de sélectionner des étiquettes syntaxiques pour des séquences de mots fixes mais pour les différentes segmentations possibles (pouvant s'entrecouper) d'une phrase,

indépendamment les unes des autres. La première étape consiste à estimer les probabilités des étiquettes syntaxiques en fonction des mots. Pour cette tâche, nous choisissons d'employer une méthode supervisée de classification binaire. La deuxième étape est la sélection des étiquettes. Chaque mot possible est associé, à partir du lexique, à un ensemble de classes grammaticales. On peut donc sélectionner les étiquettes soit pour les segments de texte soit pour les couples mot/classe grammaticale, que l'on nomme ici « entrées lexicales ». La distinction permet d'établir deux modes de sélection des étiquettes syntaxiques permettant de sélectionner conjointement les classes grammaticales et les étiquettes syntaxiques des mots.

Par ailleurs, dans ces travaux, nous décidons d'une part de nous limiter à la sélection des noms de dépendances et non des groupes qui sont moins précis et ne réduisent pas suffisamment l'ambiguïté. D'autre part, les classes grammaticales que nous considérons sont celles de la CDGFr, c'est à dire ni les classes grammaticales générales, ni les classes grammaticales étendues vues dans la section 5.2 mais les 185 classes grammaticales de la grammaire, comportant des informations grammaticales très précises.

• **La méthode de classification** que nous proposons, avant l'étape de sélection des étiquettes syntaxiques, vise à calculer les probabilités des étiquettes syntaxiques en fonction des mots et de leurs possibles classes grammaticales. Dans un premier temps, à partir du lexique et de la grammaire CDGFr, l'analyseur produit pour une phrase donnée tous les triplets ⟨mot, classe grammaticale, étiquette syntaxique⟩ possibles, c'est à dire tous les regroupements de tokens lexicalement valides avec chacune des classes grammaticales et chacune des étiquettes syntaxiques qui peuvent leurs être associées.

Par exemple dans une phrase telle que « Pénélope écrit des bandes dessinées engagées. » le segment de texte « bandes dessinées » est segmenté de deux façons (« bandes dessinées » ou « bandes » et « dessinées ») puis chaque segmentation est associée à différentes classes grammaticales (voir tableau 5.6) et chaque classe grammaticale à différentes étiquettes syntaxiques. Ainsi, 50 triplets différents sont définis à partir du segment « bandes », 103 pour « dessinées » et 34 pour « bandes dessinées ».

Pénélope	/ N(Lex=proper)
écrit	/ Adj(F=modifier)
écrit	/ N(Lex=common)
écrit	/ Vt(F=pz,C=a,T=past)
écrit	/ V2t(F=pz,C1=a,C2=d|g|l,T=past)
écrit	/ V2t(F=pz,C1=d,C2=g|inf,T=past)
écrit	/ V2t(F=fin,C1=d,C2=inf)
écrit	/ V2t(F=fin,C1=a,C2=d)
des	/ Conj(Lex!=ni,F=aggr)
des	/ Det(Lex=art,C=p)
des	/ Det(Lex=art|pn)
des	/ PP(F=circ|att)
des	/ PP(F=select)
des	/ PP(F=compl-Agent)
des	/ PP(F=compl-obl,C=o)
des	/ PP(F=compl,C=g|p)
bandes	/ N(Lex=common)
bandes	/ Vt(F=fin,C=a)
bandes dessinées	/ N(Lex=common)
dessinées	/ Adj(F=modifier)
dessinées	/ Vt(F=pz,C=a,T=past)
engagées	/ Adj(F=modifier)
engagées	/ N(Lex=common)
engagées	/ Vt(F=pz,C=a,T=past)
engagées	/ V2t(F=pz,C1=a,C2=d|g|l,T=past)
.	/ FullStop

TABLE 5.6 – Les entrées lexicales (i.e. couples mot/classe grammaticale) possibles pour la phrase « Pénélope écrit des bandes dessinées engagées . ».

Dans un deuxième temps, nous choisissons d'employer la méthode MaxEnt (Ratnaparkhi, 1996) via le logiciel Wapiti (Lavergne *et al.*, 2010) pour estimer les probabilités des étiquettes syntaxiques de chaque triplet en fonction du mot et de la classe grammaticale auxquels elles sont associées.

La méthode consiste à classifier les triplets en « 1 » ou « 0 », signifiant que l'étiquette syntaxique est probable ou non en fonction des informations données, et d'y joindre sa probabilité. L'apprentissage se fait alors sur tous les triplets possibles des phrases d'entraînement provenant du CDG Treebank, étiquetés « 1 » s'ils sont corrects pour la phrase et « 0 » sinon. On considère qu'un triplet correct est un triplet pour lequel le mot correspond à une segmentation correcte de la phrase et la classe grammaticale et l'étiquette syntaxique sont correctes pour ce mot.

Le mode MaxEnt de wapiti exploite le même système de patrons de traits pour l'apprentissage que dans le cas des CRF. Comme dans la section précédente, nous choisissons ici des traits classiques (e.g. mot et classe grammaticale). En outre, les entrées lexicales étant particulièrement précises, nous disposons également d'informations telles que le lemme ou le mode et le temps des verbes. Par ailleurs, la segmentation n'étant pas connue, il n'est pas possible de tenir compte des mots précédents et suivants lors de la classification. Les traits choisis sont présentés dans le tableau 5.7.

	Traits
sur les mots	w/l, $lemme/l$
sur les classes grammaticales	c/l, $c/mode + temps/l$ (pour les verbes)
autres	$sorte\ d'arc/l$

TABLE 5.7 – Patrons de traits pour la classification des triplets. Pour un triplet donné $\langle w,c,l\rangle$, w correspond au mot, c à sa classe grammaticale et l à l'étiquette syntaxique. Chaque trait est concaténé avec l'étiquette syntaxique l. La *sorte* de l'arc (dépendance) arrivant sur le mot est « projectif » ou « ancre » (il n'est pas utile de préciser « non-projectif » car chaque dépendance non-projective est couplée à exactement une ancre, i.e. un mot recevant une dépendance non-projective reçoit également une ancre).

• **La pré-sélection** des étiquettes syntaxiques (via la classification des triplets) avant analyse peut se faire selon deux modes différents :

- *mode 1*, sélection des étiquettes les plus probables pour chaque entrée lexicale (i.e. couples mot/classe grammaticale) ;
- *mode 2*, sélection des étiquettes les plus probables pour chaque segment de texte (indépendamment de la classe grammaticale).

Le premier mode permet d'obtenir la liste ordonnée des étiquettes syntaxiques selon leurs probabilités pour chaque entrée lexicale d'une phrase donnée en entrée de l'analyseur. Tandis que le deuxième mode regroupe les listes des couples correspondant au même segment de texte (i.e. même mot) sans tenir compte des classes grammaticales. En exemple, les entrées lexicales possibles pour la phrase « Pénélope écrit des bandes dessinées engagées . » sont présentés dans le tableau 5.6. Le but de l'étape de sélection est de réduire le nombre d'étiquettes des listes. Pour les deux modes, les listes d'étiquettes peuvent être élaguées selon des critères variables. Le deuxième mode permet alors de réduire plus fortement le nombre d'étiquettes sélectionnées au total sur la phrase car le premier mode conserve toutes les possibilités de classes grammaticales de chaque mot alors que le deuxième mode peut en même temps réduire le nombre de classes grammaticales à prendre en compte lors de l'analyse pour chaque mot.

Par exemple, pour le mot « bandes » il est possible de sélectionner une seule étiquette (la plus probable : **o-obj**, correspondant au triplet ⟨bandes, N(Lex=common), o-obj⟩) avec le deuxième

mode tandis qu'avec le premier mode le nombre minimal de triplets sélectionnés serait de deux car le mot appartient à deux classes grammaticales, donc l'étiquette la plus probable pour chaque entrée lexicale serait choisie (e.g. **o-obj** pour le triplet ⟨bandes, N(Lex=common), o-obj⟩ et **S** pour le triplet ⟨bandes, Vt(F=fin,C=a), S⟩).

Les triplets peuvent être retirés des listes selon leurs poids ou leurs rangs. Le poids p_t de chaque triplet t est calculé par rapport à la probabilité $P(t) \in [0,1]$[5] du triplet, soit :

$$p_t = \begin{cases} 0 & \text{si } P(t) < 0{,}0001 \\ 10\,000 - \frac{log(P(t))}{log(0{,}0001)} * 10\,000 & \text{sinon} \end{cases}$$

Chaque étiquette obtient en conséquence un poids compris entre 0 et 10 000[6]. L'élagage des listes est alors effectué selon deux critères :

- le rang $r \in [1,30]$, permettant de sélectionner les r triplets les plus probables pour chaque segment de texte ou entrée lexicale ;
- le poids p_t, permettant de sélectionner seulement les triplets pour lesquels $p_t \geq p_{max} - \delta$ où p_{max} est le score maximum (i.e. le poids du triplet le plus probable dans la liste) et $\delta \in [0,9999]$ est un paramètre défini avant la sélection.

Les résultats de l'évaluation de l'étape de classification sont présentés conjointement aux résultats sur l'analyse en dépendance dans la section qui suit.

5.3.2 Analyse en dépendances

Nous souhaitons évaluer la pertinence de la sélection automatique des triplets sur l'analyse en dépendances dans le but d'estimer la performance du processus global autonome d'analyse en dépendances. L'étape d'analyse en dépendances est donc effectuée en conformité avec les règles de la CDGFr et restreinte par la pré-sélection automatique des triplets précédemment décrite.

• **La procédure d'analyse** comprend une première étape d'affectation des types aux mots des phrases données (voir le chapitre 4 pour la définition des types et la procédure d'analyse du CDG Lab). Les triplets sélectionnés lors de l'étape précédente permettent de restreindre cette affectation. À chaque mot possible d'une phrase donnée est attribué un ensemble de types correspondant aux triplets sélectionnés pour ce mot, i.e. pour un triplet $\langle w,c,l \rangle$, tous les types de la grammaire dont la tête correspond à l'étiquette l et appartenant à la classe grammaticale c sont attribués au mot w. Ces types sont placés dans une matrice triangulaire dont la longueur correspond aux nombres de mots possibles de la phrase. La seconde étape de l'analyse en dépendances consiste à remplir la matrice en calculant toutes les combinaisons possibles entre les différents types affectés aux mots et d'en extraire les structures de dépendances possibles pour la phrase donnée.

[5] Si le triplet est étiqueté « 1 » alors $P(t) = p$ la probabilité associée donnée par Wapiti, si le triplet est étiqueté « 0 » alors $P(t) = 1 - p$.

[6] Le poids 0 correspond à un triplet dont la probabilité est inférieure à 0,0001 et le poids 10 000 à un triplet dont la probabilité est 1.

• **Les expérimentations** sont effectuées sur les phrases du CDG Treebank. Le corpus est divisé en 10 parties équivalentes qui sont chacune traitées indépendamment les unes des autres. Chaque étape de classification comprend une phase d'apprentissage sur 90 % du corpus et une phase de classification et de pré-sélection sur les 10 % restants. Ces 10 %, dont chaque phrase est associée à une sélection de triplets, sont donnés à l'analyseur pour effectuer l'analyse en dépendances.

Les évaluations consistent à comparer les résultats d'une part de l'étape de classification des triplets et d'autre part de l'étape d'analyse en dépendances, en fonction des différents modes et critères de sélection. L'objectif est d'évaluer la pertinence de ces différents critères de sélection des triplets dans le but de réduire suffisamment les listes de triplets sans sacrifier la précision des analyses, c'est à dire en conservant les bonnes classes grammaticales et étiquettes syntaxiques des mots parmi les triplets sélectionnés. Les critères de sélection des triplets sont le rang et le poids. Nous choisissons les valeurs 1, 2, 5 et 10 pour effectuer des expérimentations avec une limitation sur le rang et les valeurs 100, 200, 500, 1000, 2000 et 3000 pour le paramètre δ permettant de limiter les sélections en fonction du poids des triplets (voir la section 5.3.1).

L'évaluation de l'étape de classification comprend le calcul :

- de la précision de la sélection des triplets, i.e. le pourcentage de triplets corrects trouvés :

$$\frac{\text{nombre de triplets sélectionnés corrects}}{\text{nombre de triplets corrects dans l'ensemble d'origine}}$$

- du nombre moyen de triplets sélectionnés par mot.

Pour l'évaluation de l'étape d'analyse en dépendances, nous choisissons de calculer les scores à partir de la meilleure structure de dépendances de chaque phrase, c'est à dire la structure ayant le plus de dépendances en commun avec la structure d'origine dans le corpus parmi les analyses produites par l'analyseur. Dans le calcul des résultats globaux nous prenons en compte également les analyses non abouties (par non-conformité avec la grammaire ou par dépassement de la limite de temps de 10 secondes). Les analyses non abouties produisent des structures vides, i.e. aucune dépendance n'est correcte. Nous calculons d'une part le LAS (*labelled attachement score*) et le UAS (*unlabelled attachement score*) (voir section 5.2.2) et d'autre part, le temps de calcul moyen des analyses (en seconde) par phrase.

• **Les résultats** des évaluations croisées sont présentés dans les tableaux 5.8 et 5.9. Le tableau 5.8 présente les résultats des expérimentations avec le mode de sélection des triplets par entrée lexicale, i.e. les triplets sont sélectionnés pour chaque couple mot/classe grammaticale de chaque phrase (toutes les classes grammaticales possibles sont donc conservées lors de l'analyse). On observe que le nombre moyen de triplets sélectionnés par mot est au minimum de 5 et au maximum de 31,5. La précision sur la sélection des triplets atteint 98,2 % pour une sélection des 10 triplets les plus probables. Dans le cas de l'analyse en dépendances, les meilleurs scores (48,8 % en UAS et 39,8 % en LAS) sont obtenus pour $\delta = 500$.

Le tableau 5.9 présente les résultats des expérimentations avec le mode de sélection des triplets par segment de texte (avec ce mode certaines classes grammaticales peuvent ne pas être sélectionnées). Le nombre moyen de triplets sélectionnés par mot est au minimum de 1 (pour le critère $r = 1$) et ne dépasse pas 9,8 (pour le critère $\delta = 3000$). Avec ce dernier critère, la précision de la sélection atteint 91,9 % (le maximum parmi les résultats présentés pour ce mode). Mais les scores d'analyse en dépendances sont les plus élevés (60.8 % en UAS et 53.9 % en LAS) pour $r = 5$.

[7] Le nombre moyen de triplets sélectionnés par mot peut être inférieur à la valeur de r (le critère de limitation selon le rang) car il y a parfois moins de r possibilités de triplets pour un mot donné.

Critères		Résultats classification		Résultats parsing		
r	δ	**triplets/mot**	**précision**	**UAS**	**LAS**	**temps**
1	**-**	5,0	66,7	47,2	38,2	0,85
2	**-**	9,6	80,2	46,1	40,0	2,67
3	**-**	13,9	87,9	39,8	36,9	4,11
5	**-**	21,3	93,6	32,9	31,6	5,44
10	**-**	31,5	**98,2**	24,0	23,7	6,62
-	**100**	5,3	67,6	47,8	39,1	0,93
-	**200**	5,6	68,1	48,5	39,6	1,01
-	**500**	6,5	70,1	**48,8**	**39,8**	1,26
-	**1000**	8,8	75,5	46,2	39,5	2,32
-	**2000**	15,8	86,9	37,9	34,8	4,25
-	**3000**	26,1	95,6	29,7	29,0	5,83

TABLE 5.8 – Résultats des expérimentations sur la classification des triplets et l'analyse en dépendances avec le mode de sélection des triplets par entrée lexicale. r et δ sont les critères de sélection définis au préalable permettant de limiter les sélections en fonction respectivement du rang et du poids.

Globalement, pour chacune des expérimentations on constate que l'augmentation de la précision sur la sélection des triplets n'est pas toujours avantageuse pour l'analyse en dépendances. En effet, l'augmentation de la précision est due à l'augmentation du nombre de triplets sélectionnés par mot. Cependant, plus le nombre de triplets sélectionnés augmente plus l'espace de recherche lors d'une analyse est large et plus longs sont les temps de calcul. Il y a donc plus de chances que les analyses n'aboutissent pas. Ainsi les scores globaux chutent pour un trop grand nombre de sélections de triplets. Les scores des expérimentations avec le mode de sélection par entrée lexicale (tableau 5.8) sont donc relativement bas dans l'ensemble car le nombre moyen de triplets sélectionnés par mot est trop élevé. La sélection des triplets par entrée lexicale ne permet pas de réduire suffisamment l'espace de recherche des analyses. Dans le cas des expérimentations avec le mode de sélection par segment de texte, on remarque que le nombre moyen de triplets sélectionnés par mot est dans l'ensemble beaucoup plus bas et que les temps d'analyse sont également moins élevés (au maximum 3,75 secondes par phrase contre 6,62). Un espace de recherche plus restreint permet d'aboutir à plus d'analyses. Ainsi les scores d'analyse en dépendances avec le second mode de sélection devancent (+12,0 en UAS et +14,8 en LAS en considérant les scores maximaux) ceux avec le premier mode bien que la précision sur la sélection des triplets n'atteignent pas des scores aussi hauts (-6,3 sur les précisions maximales).

5.3.3 Discussion

Dans cette section nous avons proposé une méthode de pré-sélection jointe des classes grammaticales et des étiquettes syntaxiques avant analyse en dépendances. Nous avons mis en place un processus comprenant la classification des triplets ⟨mot, classe grammaticale, étiquette syntaxique⟩ pour chaque phrase à analyser, la sélection restreinte de ces triplets et l'analyse en dépendances tenant compte de cette sélection. La sélection des triplets permet de réduire l'espace de recherche des analyses en dépendances et ainsi d'obtenir plus rapidement une réponse de l'analyseur dans le cadre d'une analyse autonome (sans intervention humaine).

Nous avons proposé deux modes de sélection des triplets : en fonction des entrées lexicales et

Critères		Résultats classification		Résultats parsing		
rang	**poids**	**triplets/mot**[7]	**précision**	**UAS**	**LAS**	**temps**
1	-	1,0	46,9	33,1	29,6	0,68
2	-	1,9	56,7	44,0	37,1	0,69
3	-	2,9	67,0	56,7	47,8	0,71
5	-	4,5	77,3	**60,8**	**53,9**	1,51
10	-	8,4	87,9	44,8	42,1	3,75
-	**100**	1,2	48,2	34,6	30,8	0,67
-	**200**	1,3	49,3	36,1	32,1	0,67
-	**500**	1,5	52,9	38,6	34,0	0,67
-	**1000**	2,2	61,5	45,4	38,7	0,68
-	**2000**	4,9	79,3	57,3	50,3	1,34
-	**3000**	9,8	**91,9**	49,3	47,2	3,50

TABLE 5.9 – Résultats des expérimentations sur la classification des triplets et l'analyse en dépendances avec le mode de sélection des triplets par segment de texte.

en fonction des segments de texte. Le premier mode de sélection, conservant toutes les possibilités de classes grammaticales pour chaque mot possible de chaque phrase, ne permet pas de réduire suffisamment l'espace de recherche des analyses. Le second mode est plus efficace pour réduire le nombre moyen de triplets sélectionnés par mot bien que la précision soit réduite. Cependant, les scores d'analyses sont meilleurs grâce à la réduction des temps d'analyse qui font que plus d'analyses aboutissent en un temps réduit.

Globalement, la méthode que nous proposons permet de s'affranchir de l'intervention des annotateurs lors de l'étape de pré-annotation (en particulier la pré-sélection des étiquettes syntaxiques) des phrases. Cependant les scores d'analyse en dépendances sont relativement bas, ce qui signifie que beaucoup d'erreurs d'annotations sur les dépendances ou dans le pire des cas aucune annotation n'est fournie à l'annotateur pour la phase de correction des structures de dépendances. Par conséquent, le bénéfice serait moindre sur le temps de travail des annotateurs.

La méthode introduit certaines limitations. D'une part la réduction de l'espace de recherche est limité par le fait que toutes les segmentations possibles des phrases à analyser sont conservées. D'autres part, les scores d'analyse relativement bas sont dus à la mauvaise classification des triplets car les informations fournies lors de l'entraînement ne tiennent pas compte du contexte (également dû au fait que le contexte n'est pas fixe). Le principal problème vient donc de la segmentation incertaine des phrases lors de la classification.

5.4 Processus de pré-annotation automatique séquentiel

L'approche présentée dans la section précédente (section 5.3) ne permet pas de tenir compte d'un contexte fixe pour prédire efficacement les étiquettes syntaxiques des mots. Or ces informations sont particulièrement importantes dans le cadre d'un travail ayant une dimension syntaxique et où les informations pertinentes sont parfois distantes.

Nous proposons dans cette section une seconde méthode de pré-annotation automatique permettant d'effectuer indépendamment la segmentation des phrases en mots, l'étiquetage grammatical et l'étiquetage syntaxique de ces mots dans le but d'obtenir une meilleure sélection des étiquettes syntaxiques en amont de l'étape d'analyse en dépendances. En effet, une segmentation fixe

permet de prendre en compte des informations précises sur le contexte des mots lors de l'apprentissage de modèles statistiques. De plus, les étiqueteurs performants ne sont pas rares aujourd'hui en ce qui concerne le français. Nous souhaitons donc profiter des bonnes performances de ces outils pour la mise en place du processus de pré-annotation des phrases.

En outre, nous avons vu que la sélection automatique des étiquettes syntaxiques est efficace pour réduire l'espace de recherche des analyses mais que les erreurs d'étiquetage peuvent induire un espace dans lequel ne se trouve pas la bonne solution. Les analyses sont effectuées rapidement mais ne sont pas suffisamment précises. Il est donc intéressant de sélectionner plus d'une étiquette par mot pour élargir l'espace de recherche à de meilleures solutions admises par la grammaire. On en revient à devoir trouver un compromis entre la réduction de l'ambiguïté des analyses (et donc du temps de calcul) et la précision de ces analyses. Nous employons alors une méthode permettant de gérer des listes d'étiquettes prédites pour les mots en tenant compte de leurs probabilités.

Pour finir, nous souhaitons intégrer le processus de pré-annotation automatique dans le processus complet d'annotation du CDG Lab. La procédure complète d'annotation en dépendances que nous proposons d'intégrer au CDG Lab comprend :

- la segmentation automatique des phrases en mots ;
- la pré-annotation syntaxique automatique des mots (utilisant une étape d'étiquetage grammatical) ;
- l'analyse en dépendances dirigée par la CDGFr ;
- la validation/modification des annotations par un annotateur.

Nous présentons alors les problèmes engendrés par l'intégration de ce processus sur la validation des structures et les solutions pour y remédier. Puis nous discutons du bénéfice de la pré-annotation automatique sur le processus complet autour de l'évaluation d'un ensemble de phrases.

5.4.1 Segmentation

La procédure de segmentation consiste à découper les phrases en mots en acceptant les expressions multi-mots, afin de proposer aux annotateurs une segmentation cohérente avec le schéma d'annotation des corpus CDG.

Dans un premier temps, les phrases sont découpées en tokens, puis les associations possibles de tokens sont effectuées de manière à conserver les séquences de tokens les plus longues reconnues par le lexique associé à la CDGFr. Cette première procédure simple engendre un taux d'erreur de 2,85 % sur le CDG Treebank par rapport à la segmentation d'origine. Cependant, la plupart des erreurs sont des erreurs redondantes et classiques sur les mêmes ensembles de mots. Par exemple, certaines erreurs proviennent du mauvais traitement des articles partitifs tels que « de la ». La segmentation de base considère mécaniquement « de la » comme étant l'équivalent féminin de « du ». Ainsi, dans un second temps, une liste noire des fusions de tokens à éviter est établie dans le but de diminuer le nombre de ces erreurs fréquentes. La mise en place de cette liste noire permet de réduire le taux d'erreur à 1,62 %.

5.4.2 Étiquetage grammatical

Le CDG Treebank est entièrement annoté avec les classes grammaticales définies pour la CDG Fr. Dans la section 5.2, nous proposions des sous-classifications de cette classification permettant de

réduire l'ensemble d'étiquettes à 86 classes étendues ou 28 classes générales. Le jeu de 86 étiquettes est très précis et donc utile pour la désambiguïsation des étiquettes syntaxiques mais comporte lui-même des étiquettes avec des informations syntaxiques précises difficiles à prédire en tenant compte seulement des mots. D'autre part, le jeu de 28 classes comportent 10 étiquettes pour la ponctuation et 18 étiquettes pour les mots hors ponctuation. La disjonction de certaines classes parmi ce jeu d'étiquette ne semble pas nécessaire dans la désambiguïsation des étiquettes syntaxiques. En effet, par exemple, les ponctuations sont différentiables par leurs formes. De plus, la séparation entre certaines classes pour les verbes telles que *Vt* (verbes transitifs) et *V2t* (verbes ditransitifs) n'est pas facilement discernable automatiquement. Par ailleurs, ce jeu d'étiquettes ne correspond pas au jeu standard utilisé dans une majorité de travaux à travers la communauté française du traitement de la langue. Ainsi, de façon à pouvoir exploiter les bonnes performances d'un étiqueteur classique dans le domaine lors d'une étape d'étiquetage grammatical, nous décidons de convertir notre jeu d'étiquettes en un jeu d'étiquettes standard.

• **La conversion du jeu d'étiquettes** consiste à établir une équivalence entre l'ensemble des classes grammaticales de la CDGFr et un nouveau jeu d'étiquettes dans le but d'assigner ces nouvelles étiquettes aux mots du corpus. Nous choisissons d'utiliser le jeu d'étiquettes nommé TREEBANK+ par Crabbé et Candito (2008) jugé efficace pour l'analyse syntaxique. La plupart des étiquettes de ce jeu résultent d'une combinaison entre les étiquettes standards utilisées pour l'annotation du French Treebank (Abeillé *et al.*, 2003) et certaines informations provenant des sous-catégories de ces étiquettes. Il comprend 28 étiquettes étendues. Il s'agit d'autre part du jeu d'étiquettes employé par l'étiqueteur Melt (Denis et Sagot, 2009, 2012), largement utilisé dans la communauté et atteignant une précision de 97 % sur le français.

La plupart des classes grammaticales de la CDGFr ont une correspondance avec une étiquette TREEBANK+. Néanmoins, quelques classes telles que *Explet* (interjections), *Colloc* (collocations) et *Part* (partitifs) n'ont pas d'équivalent dans le sens où, pour une classe grammaticale donnée, les mots appartenant à cette classe dans le CDG Treebank ne correspondent pas tous à la même étiquette TREEBANK+. Il existe donc des ambiguïtés dans les choix de correspondance entre certaines classes grammaticales et les étiquettes TREEBANK+. Une conversion directe des classes vers les étiquettes TREEBANK+, forçant ces choix, engendrerait des erreurs dues à ces ambiguïtés. Nous choisissons de procéder à une conversion mixte des étiquettes grammaticales du CDG Treebank vers les étiquettes TREEBANK+, incluant deux étapes. Nous étiquetons automatiquement dans un premier temps le CDG Treebank à l'aide de l'étiqueteur Melt. Dans un second temps, nous appliquons des règles de conversion qui permettent de corriger automatiquement les mauvaises prédictions dans les cas où la classe grammaticale permet de définir sans ambiguïté l'étiquette TREEBANK+ correspondante. Il en résulte un taux d'erreur de 6 %. Les erreurs les plus fréquentes sont dues aux ambiguïtés existantes entre les adjectifs prenant le rôle de noms communs ou les participes passés prenant le rôle d'adjectifs. Un exemple de conversion est donné par le tableau 5.10.

Par ailleurs, les erreurs d'étiquetage de Melt permettent de mettre en évidence les disparités entre les phrases du CDG Treebank et du corpus d'entraînement de Melt, le French Treebank. L'un des problèmes est le manque de phrases comprenant un verbe conjugué à la seconde personne du singulier (à l'indicatif ou à l'impératif) dans le French Treebank (dû au style journalistique) comparé au CDG Treebank qui en contient un pourcentage important. Notamment, Melt n'est pas capable de trouver la majorité des verbes conjugués à l'impératif à la seconde personne du singulier. Beaucoup de ces verbes sont étiquetés comme étant des verbes à l'indicatif mais une part non négligeable est également étiquetée en tant que nom commun car apparaissant en début de phrase avec une première lettre en majuscule. Un autre exemple d'erreurs fréquentes est l'étiquetage du pronom « tu » comme verbe, considéré comme étant la forme conjuguée du verbe « taire ».

Mot	Classe grammaticale CDG	POS-tag Melt	POS-tag direct	POS-tag final
Non	Colloc(F=sent)	ADV	-	ADV
,	Comma(Lex=',')	PONCT	PONCT	PONCT
il	PN(Lex=pers,C=n)	CLS	CLS	CLS
faut	Vt(F=fin,C=a)	V	V	V
que	PN(F=subord)	CS	-	CS
tu	PN(Lex=pers,C=n)	VPP	CLS	CLS
le	PN(Lex=pn,F=clit,C=a)	DET	CLO	CLO
fasses	Vlight(F=fin)	NC	VS	VS
manger	Vt(F=inf,C=a)	VINF	VINF	VINF
.	FullStop(Lex='.')	PONCT	PONCT	PONCT

TABLE 5.10 – Exemple de conversion des étiquettes grammaticales du CDG Treebank vers les étiquettes TREEBANK+ pour la phrase « Non, il faut que tu le fasses manger. ». Certains POS-tags non-correctement assignés par Melt (en rouge) sont corrigés (en vert) par la conversion directe de certaines classes grammaticales non-ambiguës vers les étiquettes TREEBANK+. Aucune correction n'est faite à partir des classes grammaticales ambiguës, comme par exemple ici pour les mots associés aux classes Colloc(F=sent) et PN(F=subord). Les étiquettes assignées par Melt sont alors conservées (en bleu).

• **L'étiquetage grammatical** est alors effectué par l'étiqueteur Melt. Le processus de pré-annotation étant évalué à travers une validation croisée des expérimentations, chaque partie de test du corpus est automatiquement annoté par Melt tandis que les annotations des mots de chaque partie d'entraînement proviennent de la conversion du jeu d'étiquettes exposé précédemment.

5.4.3 Étiquetage syntaxique

Le but de l'étiquetage syntaxique consiste, comme expliqué en premier lieu dans la section 5.2, à prédire les étiquettes des dépendances arrivant sur les mots sans prédire les dépendances. En outre, comme précisé dans la section précédente (section 5.3), nous nous limitons également dans cette section à la prédiction des noms de dépendances et non des groupes. Dans le cadre de nos travaux, prédire les étiquettes des mots signifie prédire une ou plusieurs étiquettes par mot et procéder à une sélection plus ou moins restreinte de ces étiquettes. Nous avons montré, dans les sections précédentes, l'impact du choix du nombre d'étiquettes sélectionnées par mot sur l'efficacité globale de l'analyse en dépendances dirigée par les grammaires catégorielles de dépendances. L'enjeu de l'étape d'étiquetage syntaxique est donc de sélectionner un nombre d'étiquettes par mot suffisant pour obtenir la bonne étiquette parmi celles-ci tout en limitant ce nombre dans le but de réduire les temps d'analyse en dépendances. Il s'agit donc de trouver un compromis entre précision et réduction de l'ambiguïté.

• **La méthode d'étiquetage syntaxique** que nous souhaitons mettre en place doit pouvoir prédire une liste ordonnée d'étiquettes pour chaque mot d'une phrase donnée. Les CRF sont bien adaptés à la prédiction de séquences d'étiquettes. Néanmoins, prédire les étiquettes par séquences signifie qu'une même étiquette peut être assignée plusieurs fois à un mot et que nous ne pouvons pas obtenir la probabilité d'une étiquette pour un mot donné car les CRF calculent la probabilité des séquences d'étiquettes. Nous employons alors la méthode MaxEnt permettant de prédire des étiquettes et leurs probabilités pour les mots indépendamment les uns des autres. Les patrons de traits permettant de définir les traits utilisés pour l'étiquetage syntaxique sont présentés dans

le tableau 5.11. Nous avons choisis uniquement d'extraire des traits sur les mots et les POS-tags (i.e. les étiquettes grammaticales du jeu TREEBANK+, voir la sous-section précédente) car des informations redondantes par rapport à celle utilisées pour l'étiquetage grammatical ne sont pas pertinentes dans le cas de l'étiquetage syntaxique. En outre, les traits sur les POS-tags peuvent concerner des mots distants par rapport au mot courant (-5 ou +5 de distance avec le mot) car les relations syntaxiques concernent relativement souvent des mots distants (en particulier dans le cas des dépendances projectives).

Traits	
sur les mots	$w_{\{-3,3\}}$
sur les POS-tags	$t_{\{-5,5\}}$
mixtes	w/t, $t_{-1}/t_0/t_1$

TABLE 5.11 – Patrons de traits pour l'étiquetage syntaxique. w_0 correspond au mot courant et t_0 à l'étiquette (POS-tag) TREEBANK+ (résultant de la conversion du jeu d'étiquette pour les données d'entraînement et assignée par Melt pour les données de test) pour le mot courant.

Par cette méthode d'étiquetage, chaque mot reçoit une liste d'étiquettes ordonnée en fonction de la probabilité de ces étiquettes et préalablement épurée des étiquettes incohérentes avec la CDGFr[8]. Nous mettons en place une procédure d'élagage des listes en fonction des probabilités. L'idée est de supprimer de la liste les étiquettes les moins probables en instaurant des paliers proportionnels aux probabilités maximum. Ainsi, dans la liste de chaque mot du corpus, sont éliminées les étiquettes ayant une probabilité inférieure à $\alpha.p_{max}$ où p_{max} est la probabilité de la meilleure étiquette et $\alpha \in [0,1]$ est défini lors de la procédure.

• **Les expérimentations** sont effectuées sur le CDG Treebank. Pour ces expérimentations, la segmentation correcte des phrases en mots est conservée. Les évaluations sont des évaluations croisées en 10 strates. Chaque partie d'entraînement du corpus est annotée grammaticalement selon la méthode de conversion présentée dans la sous-section précédente (5.4.2) et chaque partie de test est pré-annotée à l'aide de l'étiqueteur grammatical Melt (Denis et Sagot, 2009, 2012).

L'évaluation de la méthode d'étiquetage passe tout d'abord par une comparaison avec deux méthodes classiques d'analyse en dépendances. Le calcul de la précision au rang 1 d'une méthode d'étiquetage est équivalent au calcul du *label accuracy* (LA, i.e. précision sur les étiquettes) sur les méthodes d'analyse en dépendances. Il s'agit de calculer le pourcentage de mots pour lesquels la bonne étiquette a été prédite (la plus probable dans le cas de l'étiquetage syntaxique). Nous comparons les performances, au niveau de l'étiquetage, de notre méthode avec deux méthodes d'analyse en dépendances (gérant les dépendances non-projectives et donc ignorant les dépendances ancres) dirigées par les données : la méthode *covnonproj* du MaltParser (Nivre *et al.*, 2006a) permettant d'obtenir les meilleurs scores sur nos données avec des patrons de traits optimisés pour la prédiction des étiquettes syntaxiques à l'aide du MaltOptimizer (Ballesteros et Nivre, 2012) et la méthode basée sur les graphes du MateParser de Bohnet (2010). Nous différencions, de plus, les dépendances projectives des dépendances non-projectives. C'est à dire que la précision est calculée

[8]Nous disposons grâce à la CDGFr de la correspondance entre les mots et les étiquettes syntaxiques, par l'intermédiaire d'une correspondance avec les classes grammaticales. En effet, chaque mot peut-être associé à une liste de classes grammaticales à l'aide du lexique français du CDG Lab (incluant des informations du Lefff) et chaque classe grammaticale peut être associée à une liste d'étiquettes syntaxiques (i.e. les noms de dépendances) à l'aide de la grammaire. Par transitivité, chaque mot peut donc être associé à une liste d'étiquettes syntaxiques possibles. La liste produite par la méthode MaxEnt est donc tout d'abord élaguée des étiquettes ne se trouvant pas dans la liste donnée à l'aide de la grammaire dans le but d'éviter une incompatibilité avec la grammaire.

d'une part pour les mots recevant une dépendance projective dans la structure de dépendances correcte d'origine et d'autre part pour les mots recevant une dépendance non-projective.

Par ailleurs, l'intérêt d'utiliser une méthode de classification est d'obtenir une liste d'étiquettes ordonnées selon leurs probabilités pour chaque mot dans le but de réduire les listes d'étiquettes et ainsi d'évaluer l'intérêt d'un compromis entre réduction du nombre d'étiquettes par mot et précision de l'étiquetage. Nous testons donc différentes valeurs dans l'intervalle $[0,1]$ pour α, le paramètre d'élagage et calculons la précision de l'étiquetage à différents paliers. Dans ce cas, pour chaque test, la précision correspond au pourcentage des mots pour lesquels la bonne étiquette est trouvée parmi la liste d'étiquettes élaguée.

• **Les résultats** de l'étiquetage syntaxique au rang 1 comparés aux résultats de la précision sur les étiquettes (LA) des analyses en dépendances effectuées avec le MaltParser et le MateParser sont présentés dans le tableau 5.12 [9]. Nous remarquons que notre méthode obtient de meilleurs résultats que le MaltParser sur les mots recevant des dépendances projectives (+ 1,3) mais de moins bons qu'avec le MateParser (- 0,7). Néanmoins, les scores de notre méthode sur les dépendances non-projectives sont largement supérieurs aux scores obtenus avec les deux autres méthodes (+ 9,4 par rapport au MaltParser et + 8,8 par rapport au MateParser), ce qui permet d'obtenir un score global supérieur à celui du MaltParser (+ 1,7) et équivalent par rapport au MateParser. Les résultats montrent aussi les scores obtenus sur les phrases, dont les scores sur les phrases projectives et les phrases non-projectives. Les dépendances non-projectives ne représentent pas un fort pourcentage sur l'ensemble des dépendances mais sont présentes dans beaucoup de phrases (40 %), ce qui influe notablement les résultats globaux. Cependant, ces résultats globaux sur les phrases sont très proches entre les trois méthodes (+ 0,1 pour notre méthode par rapport au MaltParser et + 0,3 par rapport au MateParser).

	Mots			**Phrases**		
	Toutes	**Proj.**	**Non-proj.**	**Toutes**	**Proj.**	**Non-proj.**
MaltParser	83,0	83,6	69,2	24,3	26,3	21,7
MateParser	84,7	85,6	69,8	24,1	28,7	18,4
Notre méthode	84,7†	84,9†	78,6‡	24,4	26,4	22,1

TABLE 5.12 – Évaluation de l'étape d'étiquetage syntaxique. Le symbole † signifie que les scores de notre méthode sont significativement supérieurs (à 0,1 % conformément au test de Student) aux scores obtenus par le MaltParser et ‡ signifie que les scores sont significativement supérieurs aux deux autres méthodes.

Une question intéressante est de noter l'intérêt de l'étiquetage sur la non-projectivité. Dans le cas des trois méthodes, l'étiquetage atteint une précision moins élevée sur les mots recevant une dépendance non-projective que sur les mots recevant une dépendance projective. Néanmoins, notre méthode atteint des scores bien plus élevés que les deux analyseurs sur les mots originellement attachés à des dépendances non-projectives. Cependant, du fait du faible pourcentage de dépendances non-projectives sur le total des dépendances (3,8 %), le score global n'en est que faiblement affecté. Dans un cas plus général, les scores plus bas atteints par les dépendances non-projectives

[9] Les scores n'atteignent pas ceux obtenus dans les travaux d'analyse en dépendances projective du français qui arrive à environ 88 % de précision. En effet, ces scores ne sont pas comparables à cause de la constitution des corpus exploités dans les travaux standards. Ces corpus en dépendances, en plus d'être projectifs contiennent un ensemble plus large de phrases pour l'entraînement que l'ensemble du CDG Treebank et utilise un jeu d'étiquettes syntaxiques réduit.

sont dus au fait qu'une grande partie de ces dépendances sont des dépendances distantes, c'est à dire que le gouverneur est potentiellement très éloigné du dépendant. En ce cas, il est plus difficile de généraliser (à travers les traits) le contexte de ces dépendants. En effet, leur contexte syntaxique est beaucoup plus variable que dans le cas de dépendances locales. C'est le cas des dépendances de type *aggr* (aggrégation) ou *copred* (co-prédication).

Par ailleurs, si on s'intéresse aux scores obtenus en fonction des différentes étiquettes syntaxiques, on remarque sans surprise que les meilleurs scores sont atteints par les étiquettes les plus fréquentes. Celles-ci couvrent les fonctions syntaxiques les plus générales telles que *pred* (sujet), *a-obj* (objet accusatif), *det* (déterminant), *modif* (modificateur) ou *prepos-g* (préposition génitive). Alors que les étiquettes syntaxiques les moins fréquentes décrivent des rôles particuliers étant relativement souvent des sous-catégories de fonctions plus générales telles que les objets de verbes copules, les verbes compléments d'auxiliaires ou les compléments des verbes (i.e. les objets). Il y a 34 étiquettes apparaissant moins de 20 fois dans le corpus, ce qui représente presque un tiers des étiquettes. Ces étiquettes sont rarement prédites dans les premiers rangs de l'étiquetage syntaxique. C'est pourquoi il est intéressant de considérer différents paliers d'élagage des listes d'étiquettes pour trouver les étiquettes rares aux rangs inférieurs. La figure 5.4 présente les résultats de l'élagage des listes. Ces résultats mettent en avant la progression de la précision en fonction du nombre moyen d'étiquettes assignées par mot après élagage. Les évaluations sont effectuées pour un α variant de 1 à 5.10^{-5}. Lorsque $\alpha = 1$, seule la première étiquette est conservée. Diminuer la valeur de α permet d'accepter plus d'étiquettes et ainsi d'accroître la précision sur les mots. Le meilleur compromis semble être pour $\alpha = 0{,}02$, admettant une moyenne d'environ 2 étiquettes par mot pour une précision de 95,5 %. Avec un α plus petit le nombre moyen d'étiquettes par mot augmente très rapidement mais ne permet pas d'atteindre des scores beaucoup plus élevés aussi rapidement. Dans le cas contraire, un α plus grand fait fortement baisser la précision pour un gain peu important sur la réduction de l'ambiguïté.

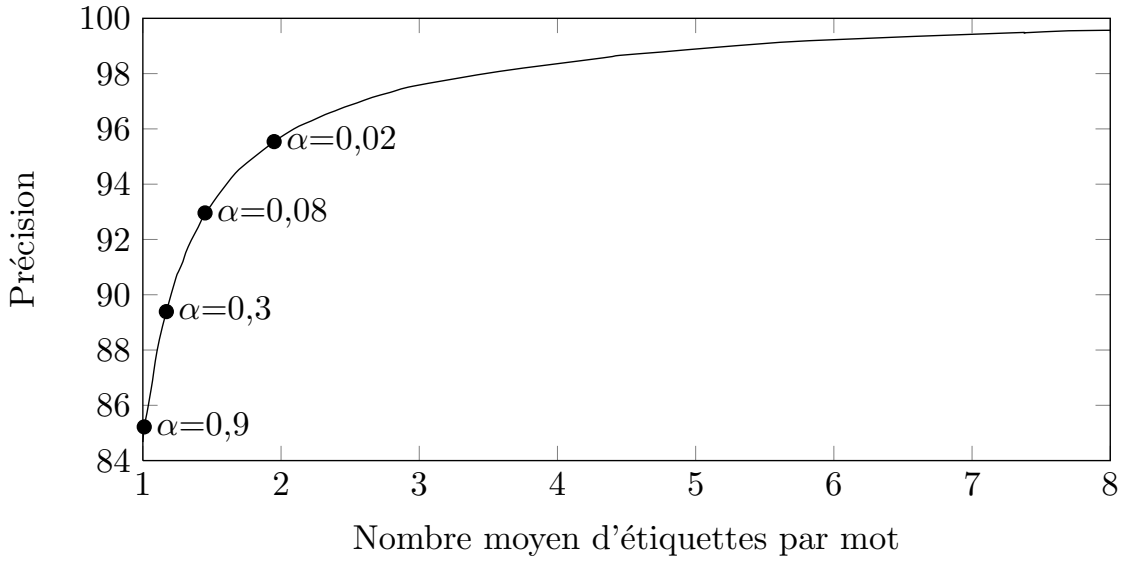


FIGURE 5.4 – Évaluation de la précision en fonction du nombre moyens d'étiquettes par mot par application de la méthode d'élagage.

5.4.4 Analyse en dépendances

Nous souhaitons évaluer l'impact du processus d'étiquetage syntaxique automatique sur l'étape d'analyse en dépendances du CDG Lab.

• **Les expérimentations** sont effectuées sur les données du CDG Treebank pré-annotées automatiquement lors de l'étape précédente d'étiquetage syntaxique pour différentes valeurs du paramètre α, conservant plus ou moins d'étiquettes par mot (comme précédemment la bonne segmentation des phrases en mots est conservée pour ces expérimentations). Les premières expérimentations sont effectuées avec les valeurs pour α indiquée dans la figure 5.4 (i.e. $0{,}9$, $0{,}3$, $0{,}08$ et $0{,}02$). Pour chacune de ces expérimentations α est fixe pour tout le corpus. Une expérimentation supplémentaire est effectuée avec un α variable en fonction de la longueur de la phrase à analyser. Plus la phrase est longue plus α sera restrictif. Le premier palier (le moins restrictif) commence avec $\alpha = 0{,}006$, permettant d'obtenir une précision très élevée sur les phrases courtes (<10 mots). Puis, des paliers intermédiaires sont définis pour différentes longueurs de phrases (voir l'annexe C) jusqu'au dernier palier avec $\alpha = 0{,}9$ pour les plus longues phrases (>50 mots). L'objectif est de mettre en place une procédure de pré-annotation des phrases (pré-annotation syntaxique et analyse en dépendances) qui soit assez rapide et précise dans le but d'accélérer et d'alléger le travail des annotateurs.

Les évaluations sont effectuées sur les meilleures structures de dépendances en sortie de l'analyseur. Cela signifie que parmi les analyses produites par l'analyseur (si plus d'une analyse est générée pour une phrase donnée), seule la structure ayant le plus de dépendances en commun avec la structure d'origine dans le corpus est prise en compte. Les analyses qui n'aboutissent pas sont également prises en compte[10]. Les critères des évaluations sont le LAS, le UAS et le temps de calcul moyen d'une analyse (en seconde par phrase).

• **Les résultats** des analyses en dépendances sont présentés dans le tableau 5.13. On y retrouve les valeurs du paramètre α utilisé lors de l'élagage des listes pour chaque expérimentation et ainsi le nombre moyen d'étiquettes sélectionnées par mot en amont de l'analyse en dépendances. La première expérimentation (pour α=0,9) montre que pour un nombre réduit d'étiquettes syntaxiques (en moyenne 1,01) sélectionnées par mot, l'analyse en dépendances peut atteindre une précision intéressante de 77,6 % en LAS (sachant qu'avec un analyseur standard tel que le MaltParser les scores en LAS sont de 78,7 % au mieux sur nos données). De plus, le temps d'analyse par phrase est faible (0,3 secondes par phrase) et donc praticable dans le cadre d'un processus d'annotation en dépendances. Les expériences suivantes (avec un α fixe) montrent également qu'il est possible d'atteindre des scores d'analyse meilleurs en acceptant un nombre plus important d'étiquettes par mot mais que les temps d'analyse augmentent rapidement (7,2 secondes par phrase avec α=0,02 pour atteindre 91,9 % en LAS). Définir des paliers progressifs permet de trouver un compromis entre un temps d'analyse praticable (3,0 secondes par phrase) et une bonne précision (90,2 % LAS) sur l'ensemble du corpus.

Dans le cas des analyses avec un α fixe, les temps de calcul sont corrects pour les phrases courtes mais explosent pour les phrases longues dès lors que le nombre d'étiquettes par mot devient trop grand. L'utilisation d'un α progressif est alors une alternative intéressante qui permet d'augmenter les scores d'attachement et de réduire les temps d'analyse pour les phrases les plus longues. D'une part, il y a plus de chances d'obtenir les bonnes analyses pour les phrases courtes. D'autre part, il y a plus de chance d'obtenir au moins une analyse en un temps correct pour les phrases longues.

[10] Si aucune structure de dépendances n'est produite lors d'une analyse, on considère lors de l'évaluation que toutes les dépendances de la structure sont incorrectes.

Paramètre d'élagage		**Nb étiquettes/mot**	**Scores**		**Temps** (sec./phrase)
			LAS	**UAS**	
α **fixe**	0,9	1,01	77,6	83,6	0,3
	0,3	1,17	81,1	86,5	0,8
	0,08	1,45	87,4	91,3	2,3
	0,02	1,95	91,9	94,6	7,2
α **progressif**		2,04	90,2	92,9	3,0

TABLE 5.13 – Résultats des évaluations de l'analyse en dépendances autonome intégrant le processus d'étiquetage syntaxique et d'élagage en amont.

5.4.5 Intégration de l'étiquetage automatique dans le processus d'annotation en dépendances du CDG Lab

L'intégration de l'étape de pré-annotation dans le processus d'annotation du CDG Lab introduit des difficultés qui n'apparaissaient pas avec l'utilisation du formulaire de sélection des têtes. Nous rappelons que le processus complet d'annotation en dépendances comprend :

- la segmentation ;
- l'étiquetage syntaxique ;
- l'analyse en dépendances ;
- la validation manuelle des structures.

Le travail des annotateurs consiste à corriger (si nécessaire) et valider les structures de dépendances produites par l'analyseur. Lors de la procédure classique, si l'analyseur propose à l'annotateur une structure de dépendances dont toutes les dépendances sont correctes, celui-ci annote positivement toutes les dépendances et ajoute la structure au corpus. Si l'analyseur propose une structure de dépendances incorrecte, l'annotateur peut annoter les dépendances positivement ou négativement et procéder à une analyse par approximation (voir la section 4.5 du chapitre 4). L'analyse par approximation consiste à relancer l'analyse en dépendances à partir de la structure annotée positivement/négativement pour trouver une meilleure solution. L'analyseur assigne des poids aux dépendances permettant de conserver les dépendances annotées positivement, d'inhiber les dépendances annotées négativement et de chercher de nouvelles dépendances en fonction des restrictions données et de la grammaire.

Cependant les étapes préliminaires dont les étapes automatiques nouvellement intégrées (segmentation et étiquetage) introduisent des erreurs et créent des obstacles dans le processus d'annotation.

« **Aucune structure de dépendances n'est proposée** » est une erreur survenant lorsque la présélection manuelle ou automatique des étiquettes syntaxiques est non-conforme avec les types de la grammaire catégorielle de dépendances du français. Si l'espace de recherche, restreint par la sélection des étiquettes syntaxiques, n'est pas compatible avec la grammaire, l'analyseur ne propose aucune solution et l'annotateur ne peut pas continuer son travail d'annotation à partir de cette solution.

Le problème est résolu dans la nouvelle version du CDG Lab grâce à la possibilité de produire des analyses partielles lorsqu'une solution complète n'existe pas. Une structure partielle est un ensemble de sous-structures correspondant à l'analyse de sous-séquences de la phrase initiale. À partir d'une structure partielle, l'annotateur peut procéder à une analyse par approximation pour trouver une structure complète. Un exemple de structure partielle obtenue en sortie de l'analyseur est présenté par la figure 5.5.

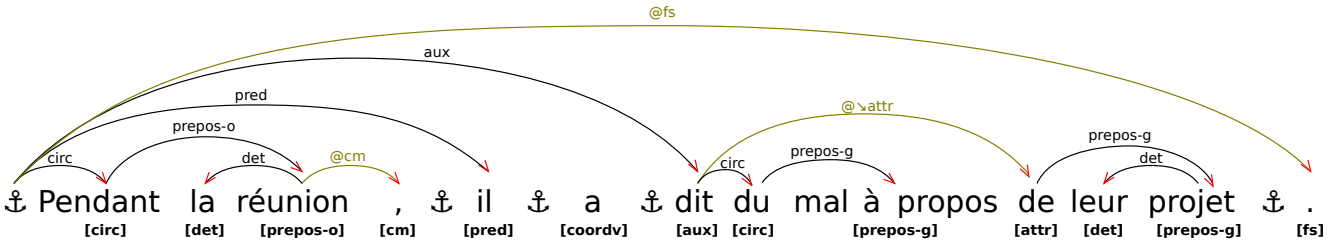


FIGURE 5.5 – Structure de dépendances partielle en sortie de l'analyseur pour la phrase « Pendant la réunion, il a dit du mal à propos de son projet. » dans le cadre du processus de pré-annotation automatique. L'analyseur a découpé la phrase en quatre sous-segments séparés par le symbole ⚓ (dont le premier représente la racine artificielle de la phrase).

« Les mots ne sont pas correctement segmentés » est une erreur induite par la mauvaise segmentation automatique de la phrase. Jusqu'alors, l'annotateur pouvait choisir les mots corrects (i.e. la bonne segmentation de la phrase) par le moyen du formulaire de sélection des têtes et n'avait pas les moyens de les modifier par la suite. Dans le cas de la pré-annotation automatique, l'annotateur n'a pas d'emprise quant à la première segmentation de la phrase et doit donc pouvoir la corriger à partir de la structure de dépendances proposée.

L'analyse par approximation s'étend donc maintenant à la correction des segments de phrases. Pour un groupe de tokens, si plusieurs segmentations existent dans le lexique de la grammaire alors ces différentes segmentations sont proposées à l'annotateur à l'aide d'un menu contextuel. L'annotateur peut alors choisir une nouvelle segmentation et lancer une analyse par approximation dans le but d'obtenir une structure prenant en compte ce nouveau découpage. la figure 5.6 présente un exemple de choix d'une nouvelle segmentation.

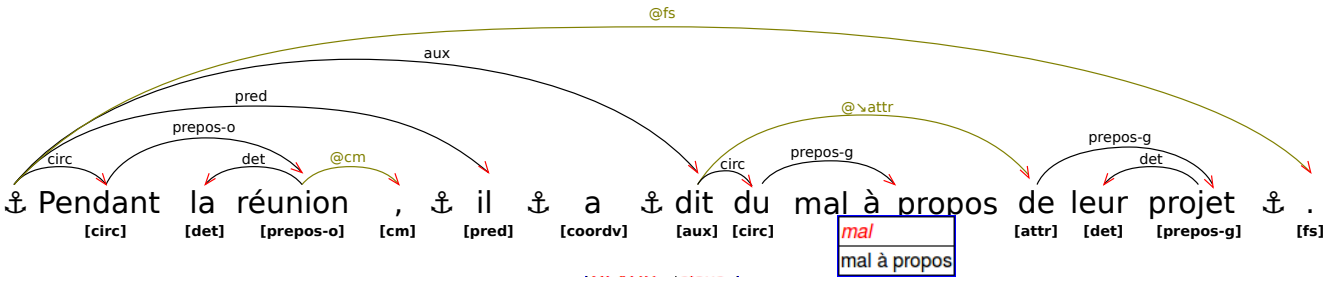


FIGURE 5.6 – Structure de dépendances en sortie de l'analyseur comprenant une erreur de segmentation pour la phrase « Pendant la réunion, il a dit du mal à propos de son projet. » dans le cadre du processus de pré-annotation automatique. Le segment de texte « mal à propos » est considéré comme une unité alors qu'il devrait s'agir pour cette phrase de trois mots distincts.

« Les étiquettes sélectionnées ne sont pas correctes » est une erreur induite par le mauvais étiquetage syntaxique des mots. L'étape de validation proposait seulement l'annotation positive ou

négative des dépendances sans différencier le cas où seule l'étiquette est incorrecte du cas où toute la dépendance est incorrecte.

Le problème est résolu par l'insertion d'une ligne superposée au texte indiquant les étiquettes de dépendances assignées à chaque mot par l'étape de pré-annotation automatique. Un menu contextuel est associé à chaque mot de la phrase, comprenant la liste des étiquettes possibles pour ces mots. L'annotateur peut alors corriger les étiquettes associés à chacun des mots indépendamment de la correction des dépendances (celles-ci peuvent toujours être annotées positivement ou négativement par les annotateurs). Lors de l'analyse par approximation, l'analyseur prend en compte les étiquettes corrigées et les étiquettes préalablement assignées pour calculer une nouvelle solution dans cet espace de recherche restreint. La figure 5.7 présente un exemple de choix d'une nouvelle étiquette syntaxique pour un mot donné d'une phrase analysée.

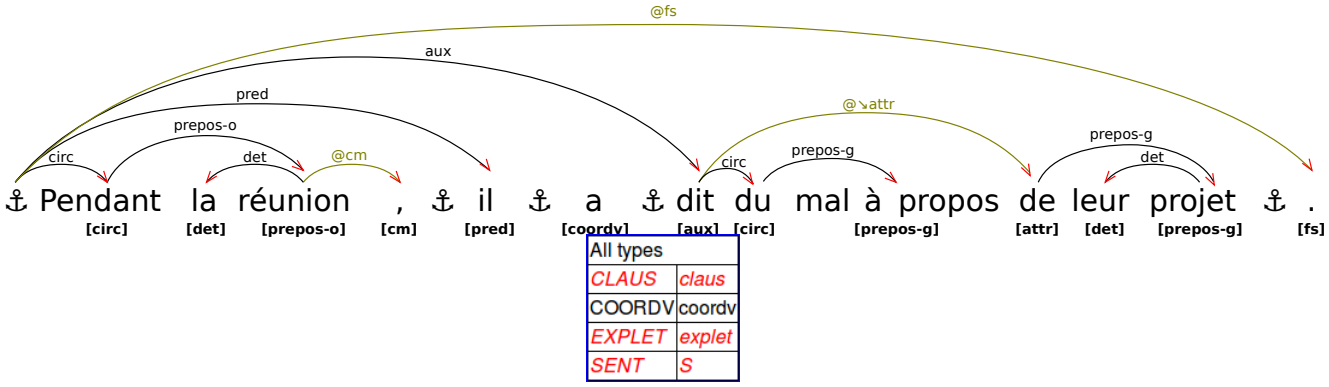


FIGURE 5.7 – Structure de dépendances en sortie de l'analyseur comprenant des erreurs d'étiquetage syntaxique pour la phrase « Pendant la réunion, il a dit du mal à propos de son projet. » dans le cadre du processus de pré-annotation automatique. Le verbe (racine) de la phrase « a » a été étiqueté *coordv* lors de l'étape d'étiquetage syntaxique. Après analyse l'annotateur a la possibilité d'ouvrir un menu contextuel pour changer l'étiquette associée à ce mot. Seules les étiquettes possibles pour ce mot sont proposées.

5.4.6 Discussion sur le processus d'annotation

Dans le but d'estimer le bénéfice de la pré-annotation automatique sur le processus complet d'annotation en dépendances dans le cadre de la construction de corpus en dépendances nous proposons d'annoter un petit ensemble de phrases via deux méthodes. La première méthode emploie le processus automatique que nous avons mis en place et la seconde méthode consiste à utiliser le formulaire de sélection des têtes pour procéder à l'annotation en dépendances des phrases. Les phrases sont extraites aléatoirement des sous-corpus inclus dans le corpus Sequoia (Candito et Seddah, 2012a). Néanmoins dans un souci d'équité, les annotations sont effectuées sur des phrases équivalentes (de même taille).

L'annotation des phrases montre que notre méthodologie est plus appropriée au travail des annotateurs. Le remplissage du formulaire de sélection des têtes est fastidieux et pas toujours intuitif du fait de devoir choisir les étiquettes des dépendances sans les dépendances elles-mêmes. L'avantage de notre méthode est de pouvoir s'affranchir du remplissage du formulaire pour parvenir directement à l'annotation de la structure de dépendances. Les annotateurs ont seulement besoin de valider les structures. Le bénéfice de notre méthode est concret sur les phrases de taille moyenne et courte (<35 mots) mais moindre sur les phrases longues. Cependant, globalement, le

temps moyen préservé avec notre méthode est environ la moitié du temps nécessaire à l'annotation avec le formulaire de sélection des têtes.

De plus, l'évaluation du processus d'annotation a permis de mettre en évidence la présence de dépendances non-projectives qui ne sont pas prises en compte dans le schéma d'annotation en dépendances du Sequoia[11]. En effet, le schéma d'annotation induit par les CDG permet de révéler un certain nombre de relations non-projectives dans les langues. Parmi les phrases que nous avons annotées, 28 % comprennent au moins une dépendance non-projective.

5.5 Conclusion

Dans ce chapitre nous avons présenté la progression des travaux nous ayant permis de valider et de mettre en place un mécanisme complet de pré-annotation dans le processus d'annotation en dépendances du CDG Lab. Nous avons montré dans un premier temps l'utilité de l'étiquetage syntaxique pour l'analyse en dépendances avec les CDG, en terme de précision et de rapidité d'analyse. À travers cette procédure, nous avons révélé la pertinence d'une méthode locale de prédiction des étiquettes de dépendances.

Ces premiers travaux, exploitant des données correctement annotées (i.e. segmentées et étiquetées), nous avons choisi d'explorer dans un second temps une méthode permettant de gérer les différentes possibilités de segmentation et d'étiquetage grammatical des phrases. Cette méthode conserve les ambiguïtés dans le but de ne pas dégrader la précision des analyses i.e. conserver toutes les possibilités de segmentation et d'étiquetage grammatical pour ne pas perdre la bonne analyse. Cependant, cette méthode, ne prenant pas en compte le contexte des mots, n'atteint pas des scores suffisants pour obtenir un compromis intéressant entre précision et rapidité d'analyse.

Dans la dernière section, nous proposons un processus de pré-annotation des phrases comprenant la segmentation et l'étiquetage grammatical, permettant de prendre en compte le contexte des mots lors de l'étape d'étiquetage syntaxique. Ce processus permet alors d'obtenir des scores d'étiquetage intéressants associés à une réduction importante du nombre d'étiquettes par mot. Ainsi, l'insertion du processus en amont de l'analyse en dépendances permet d'atteindre une précision correcte en un temps d'analyse praticable, ce qui profite alors aux annotateurs dans le cadre du processus complet d'annotation en dépendances. En outre, l'intégration du processus a permis l'évolution du système d'annotation rendant plus confortable et intuitif l'étape de validation des structures de dépendances.

D'autre part, nous avons vu que les dépendances non-projectives sont présentes en nombre suffisant pour influencer les scores des analyses. La gestion des dépendances non-projectives est alors nécessaire dans le cadre de l'analyse en dépendances et le développement de corpus en dépendances contenant des dépendances non-projectives l'est également en conséquence, d'où la nécessité d'alléger le travail des annotateurs.

[11] Le corpus Sequoia contient quelques dépendances non-projectives annotées manuellement après conversion du corpus en constituants vers les dépendances.

III

Analyse en dépendances non-projective dirigée par les données

6

Analyseur par transition adapté à la représentation des CDG

6.1 Introduction

L'analyse en dépendances est une étape utile dans de nombreux processus du traitement automatique de la langue. La rapidité et l'efficacité de cette étape est donc nécessaire au bon fonctionnement et à la précision de ces traitements tels que la traduction, l'analyse sémantique, etc. En outre des processus tels que ceux exploités dans les systèmes de question-réponse nécessitent une bonne gestion des dépendances non-projectives. Il est donc également nécessaire que les systèmes d'analyse en dépendances prédisent efficacement les dépendances projectives autant que les dépendances non-projectives.

À travers les travaux présentés dans la partie II, nous avons exploré un domaine de l'analyse en dépendances utilisant les grammaires formelles avec l'étude particulière de l'étiquetage syntaxique appliqué aux grammaires catégorielles de dépendances. Il s'agissait donc d'intégrer l'utilisation de méthodes statistiques à un processus exploitant déjà un système à base de règles. Dans la troisième partie de cette thèse, nous souhaitons étudier un mécanisme d'analyse en dépendances uniquement dirigé par les données, c'est à dire exploitant des données correctement annotées pour l'apprentissage et la prédiction des dépendances syntaxiques. Il s'agit dans ce chapitre d'étudier un système d'analyse par transition de complexité linéaire adapté à la représentation en dépendances non-projective des CDG.

Grâce au développement de corpus en dépendances, les méthodes dirigées par les données ont pris de l'ampleur dans le domaine de l'analyse en dépendances. Parmi ces méthodes d'analyse dirigées par les données, l'analyse par transition a pris une place importante ces dernières années. Les avantages d'une telle méthode sont l'efficacité, en terme de précision, couplée à la rapidité d'analyse. Les premiers formalismes d'analyse par transition étaient adaptés à la production de structures de dépendances projectives. Aujourd'hui, différents formalismes permettent de gérer également les dépendances non-projectives en temps linéaire (par l'intermédiaire de plusieurs étapes d'analyse) ou en temps polynomial.

Dans ce chapitre nous souhaitons explorer une méthode d'analyse par transition adaptée à la représentation en dépendances induite par les grammaires catégorielles de dépendances. Cette

représentation est une représentation mixte (voir la section 4.2 du chapitre 4) incluant la gestion des dépendances projectives et non-projectives, i.e. chaque mot peut être géré en même temps par une dépendance projective et une dépendance non-projective. Nous considérons dans ce chapitre que toute structure de dépendances que nous exploitons est sous cette forme. Le mécanisme que nous introduisons dans ce chapitre, initialement proposé par Alexandre Dikovsky et Boris Karlov[1], est capable de reconnaître les trois sortes d'arcs proposés par cette représentation : les dépendances projectives, les dépendances non-projectives et les ancres. L'avantage de cette méthode est de pouvoir gérer la non-projectivité grâce à un algorithme d'analyse de complexité linéaire dont les transitions s'inspirent des règles des grammaires catégorielles de dépendances pour prédire les dépendances non-projectives à partir de la notion de valence polarisée (voir la section 4.3.2 du chapitre 4).

À travers ces travaux, nous souhaitons tirer partie de la représentation mixte des CDG et analyser les avantages et les inconvénients d'un analyseur par transition basé sur cette représentation. Nous présentons dans un premier temps le formalisme de base de l'analyseur par transition adapté à la représentation des CDG. Dans un second temps nous proposons l'intégration d'un modèle statistique, un SVM, dans le but d'optimiser la prédiction des transitions. Nous proposons en outre différentes évolutions du système couvrant les faiblesses du système de base. Et nous évaluons et comparons les performances de l'analyseur à travers ces différentes évolutions.

6.2 Système par transition

Le principe de l'analyse en dépendances par transition consiste à trouver une séquence de transitions permettant de construire une structure de dépendances correcte à partir d'une phrase donnée. Les transitions (opérations complexes) sont appliquées à des configurations (états). Un système par transition est donc équivalent à un automate dans lequel les configurations décrivent l'état de l'analyse (position dans la phrase et avancement dans la construction de la structure de dépendances) et les transitions permettent de passer d'une configuration à une autre. Procéder à une analyse équivaut à chercher un chemin dans l'automate à partir de la configuration initiale jusqu'à une configuration finale. Chercher la bonne analyse revient alors à chercher le chemin permettant d'atteindre la configuration finale contenant la bonne structure de dépendances.

Le système par transition que nous présentons dans ce chapitre résulte à l'origine de travaux conjoints entre Alexandre Dikovsky et Boris Karlov. Ce système par transition est décrit par un quadruplet $\langle T, C, c_0, C_t \rangle$ où, suivant la définition présentée dans la section 2.3.2 du chapitre 2 :

- T est un ensemble de transitions ;
- C est un ensemble de configurations ;
- $c_0 \in C$ est la configuration initiale ;
- $C_t \subset C$ est l'ensemble des configurations finales.

Nous présentons les détails des configurations et des transitions propres à ce système dans les sous-sections qui suivent. La finalité de ce système d'analyse par transition est de produire des structures de dépendances conformes à la représentation en dépendances des grammaires catégorielles de dépendances. Le système intègre donc des transitions standards, équivalentes aux transitions du système *arc-standard* (Yamada et Matsumoto, 2003), pour la gestion des dépendances projectives

[1]Ces travaux ne furent pas l'objet d'une publication. Seuls les premiers tests effectués avec ce système furent présentés par Karlov et Lacroix (2012).

et des ancres, et s'étend à la gestion des dépendances non-projectives par l'ajout de transitions particulières inspirées de la notion de dualité entre les valences polarisées et par l'adaptation en conséquence des configurations.

6.2.1 Les configurations

Une configuration représente l'état dans lequel est l'analyseur au cours d'une l'analyse. Dans le cas de notre système, une configuration décrit la position de l'analyse par rapport à la phrase, i.e. l'état des mots (déjà parcourus ou non) n'étant pas encore rattachés à d'autres par des dépendances, et l'avancement de la construction de la structure de dépendances, i.e. les dépendances ayant déjà été trouvées.

Définition 14 *Configuration*
Une configuration est un quintuplet $(\sigma, \beta, \theta, k, A)$ *où :*

- σ *est une pile sauvegardant des mots ;*
- β *est une mémoire tampon comprenant des mots ;*
- θ *est une liste de valences polarisées, nommée* potentiel *;*
- k *est un compteur ;*
- A *est un ensemble d'arcs.*

Les mots empilés sur la pile σ ont été parcourus au moins une fois, ils peuvent être les gouverneurs d'autres mots (ceux-ci ont alors déjà été traités et n'appartiennent plus ni à σ ni à β) et ne sont pas encore des dépendants. Les mots déjà traités n'apparaissent plus dans la configuration. Ces mots apparaissent alors seulement en tant que dépendants dans des dépendances appartenant à l'ensemble d'arc A. Une dépendance est décrite par (i, l, a, j) où i est l'indice du gouverneur de la dépendance, j est l'indice du dépendant, l est l'étiquette de la dépendance et a est la sorte de l'arc (projectif, non-projectif ou ancre). D'autre part, les mots enregistrés dans β sont en attente de traitement. Ils sont insérés en début d'analyse dans la mémoire tampon dans l'ordre dans lequel ils apparaissent dans la phrase, puis traités les uns après les autres (de gauche à droite) lors de l'analyse. Le potentiel θ sauvegarde les valences polarisées prédites par le système dans l'ordre de leurs prédictions. Le compteur k permet de limiter la prédiction de ces valences polarisées de telle sorte qu'un mot de la phrase ne puisse être associé qu'à au plus une valence polarisée.

Pour une phrase $W = w_1 w_2 ... w_n$, la configuration initiale du système est $([w_0], [w_1,...,w_n], [], 1, \emptyset)$ où w_0 est une racine artificielle (dans la structure de dépendances résultante tout mot dépend directement ou indirectement de w_0). Une configuration finale, quels que soient σ, θ et A', est de la forme : $(\sigma, [], \theta, n+1, A')$.

6.2.2 Les transitions

Une transition est une fonction $t : C \rightarrow C$ s'appliquant aux configurations. Notre système par transition comprend 6 transitions différentes. Elles sont présentées dans le tableau 6.1. Trois des transitions du système que nous proposons sont équivalentes aux transitions employées par la méthode *arc-standard* (voir la sous-section 2.3.2 du chapitre 2).

Transition	**Effets sur les configurations**
Local-Left(l)	$(\sigma \mid w_i, w_j \mid \beta, \theta, k, A) \Rightarrow (\sigma, w_j \mid \beta, \theta, k, A \cup \{(j, l, proj \mid ancre, i)\})$
Local-Right(l)	$(\sigma \mid w_i, w_j \mid \beta, \theta, k, A) \Rightarrow (\sigma, w_i \mid \beta, \theta, next(k,j), A \cup \{(i, l, proj \mid ancre, j)\})$
Shift	$(\sigma, w_i \mid \beta, \theta, k, A) \Rightarrow (\sigma \mid w_i, \beta, \theta, next(k,i), A)$
PutPotential(v_1,...,v_m)	$(\sigma, w_i \mid \beta, \theta, k, A) \Rightarrow (\sigma, w_i \mid \beta, \theta v_1^k ... v_m^k, k+1, A)$
Dist-Left(l)	$(\sigma, \beta, \theta_1 \swarrow l^i\ \theta_2 \nwarrow l^j\ \theta_3, k, A) \Rightarrow (\sigma, \beta, \theta_1\theta_2\theta_3, k, A \cup \{(j, l, non\text{-}proj, i)\})$
Dist-Right(l)	$(\sigma, \beta, \theta_1 \nearrow l^i\ \theta_2 \searrow l^j\ \theta_3, k, A) \Rightarrow (\sigma, \beta, \theta_1\theta_2\theta_3, k, A \cup \{(i, l, non\text{-}proj, j)\})$
Conditions	
Local-Left	$i \neq 0$
Local-Right Shift	$next(k,i) = \begin{cases} k+1 & \text{si } k = i \\ k & \text{sinon} \end{cases}$
PutPotential	$i = k$
Dist-Left	si $\swarrow l^i \nwarrow l^j$ est la paire la plus à gauche satisfaisant la condition FA
Dist-Right	si $\nearrow l^i \searrow l^j$ est la paire la plus à gauche satisfaisant la condition FA

TABLE 6.1 – Définition des transitions du système d'analyse par transition.

Les transitions équivalentes aux transitions standards *Left-Arc* et *Right-Arc* sont renommées respectivement *Local-Left* et *Local-Right* étant donné qu'elles ne gèrent que les dépendances locales c'est à dire les dépendances projectives et les ancres. Elles permettent donc de lier par une dépendance gauche ou droite les deux mots en cours de traitement, i.e. le mot du haut de la pile σ et le premier mot de la mémoire tampon β. En outre, le mot subordonné dans la dépendance est supprimé (car il ne peut plus recevoir de nouvelle dépendance), il s'agit donc du mot w_i en haut de la pile σ dans le cas de *Local-Left* et du premier mot w_j de β dans le cas de *Local-Right*. De plus, dans le cas de *Local-Left*, le mot w_i du haut de la pile est passé dans β. La transition *Shift* (transition standard) permet de supprimer le premier mot de β pour l'ajouter sur σ.

Les trois autres transitions intégrées au système permettent de gérer les dépendances non-projectives. La transition *PutPotential* ajoute une ou plusieurs valences polarisées dans le potentiel θ. Les valences polarisées correspondent aux extrémités des dépendances non-projectives, comme dans le cas des grammaires catégorielles de dépendances. Par exemple, *PutPotential*($\swarrow l$) prédit une dépendance non-projective entrante étiquetée l et *PutPotential*($\nwarrow l$) prédit une dépendance non-projective sortante étiquetée l. Ces valences sont ajoutées à la liste θ dans l'ordre de leur prédiction. En outre, la transition *PutPotential* ne peut être appliquée qu'une seule fois pour chaque mot de la phrase (le compteur k dans la configuration permet de limiter l'emploi de cette transition). Si des valences duales apparaissent dans θ, les transitions *Dist-Left* et *Dist-Right* peuvent être appliquées. *Dist-Left* ajoute une dépendance non-projective gauche étiquetée l à l'ensemble A si une paire de valences (la plus à gauche dans le potentiel) $\swarrow l \nwarrow l$ satisfaisant la condition FA apparaît dans le potentiel, i.e. si ni $\swarrow l$ ni $\nwarrow l$ n'apparaissent dans θ_1 ou dans θ_2. Les valences correspondantes sont alors retirées du potentiel. *Dist-Right* ajoute une dépendance non-projective droite étiqueté l à l'ensemble A si une paire de valences $\nearrow l \searrow l$ apparaît dans le potentiel dans les même conditions que celles définies pour *Dist-Left*.

6.3 Système de prédiction

L'analyse par transition d'une phrase donnée consiste à trouver la séquence de transitions permettant de construire la structure de dépendances correcte pour la phrase. À partir d'une configuration initiale contenant la phrase, il s'agit de prédire et d'appliquer les bonnes transitions pour obtenir une configuration finale contenant la structure de dépendances. Il y a donc d'une part le système

par transition (description formelle des opérations précédemment présentées) et le système de prédiction qui détermine les transitions à appliquer [2].

Le principe d'une analyse par transition déterministe (telle qu'employée par les algorithmes du MaltParser (Nivre *et al.*, 2006a) par exemple) réside alors dans la bonne prédiction des transitions de manière locale. C'est à dire que pour chaque configuration engendrée lors de l'analyse il est nécessaire de prédire la bonne transition indépendamment des transitions précédemment appliquées. Un seul chemin (séquence de transitions) est alors produit par l'analyseur. Cette application déterministe des systèmes par transitions permet d'effectuer des analyses rapides mais n'est pas la seule manière d'exploiter ces systèmes. Par exemple, des travaux tels que ceux de Urieli (2013) propose d'intégrer la recherche par faisceau dans un système d'analyse par transition pour explorer différents chemins lors d'une analyse.

La prédiction des transitions est couramment effectuée à l'aide de modèles statistiques. Il s'agit, lors du parcours d'une analyse par transition et pour chaque nouvelle configuration engendrée, de prédire la ou les transitions les plus probables sachant cette configuration. L'emploi de modèles statistiques requière alors une étape préalable d'apprentissage.

Dans le cadre de nos travaux, nous choisissons de mettre en place un analyseur déterministe exploitant l'efficacité des SVM (*Support Vector Machine* ou séparateurs à vastes marges, voir le chapitre 3) pour la prédiction des transitions. Une étape préliminaire consiste alors à extraire d'un corpus en dépendances des données d'apprentissage pour l'entraînement du modèle statistique. Il s'agit de convertir chaque structure de dépendances correctement annotée en une séquence de transitions permettant de reconstruire cette structure. Ces données pourront être utilisées pour apprendre les probabilités des transitions en fonction des configurations. Et les probabilités serviront à prédire les transitions lors de l'analyse. Nous présentons dans la sous-section suivante la méthode de conversion des structures de dépendances en séquences de transitions puis dans la sous-section ultérieure l'intégration d'un modèle SVM à l'analyseur utilisé pour l'apprentissage et la prédiction des transitions.

6.3.1 Conversion d'une structure de dépendances en séquence de transitions

Toute structure de dépendances peut être convertie en une séquence de transitions de manière unique. La procédure de conversion est présentée par l'algorithme 5. Il s'agit d'une procédure déterministe. En conséquence pour une phrase donnée et la structure de dépendances qui lui est associée, il existe une seule séquence de transitions permettant de construire la structure.

Pour chaque mot de la phrase donnée en entrée, les dépendances non-projectives sont traitées en premier. Alors, pour chaque dépendance non-projective arrivant ou partant du mot courant, la valence polarisée correspondante est ajoutée au potentiel par le biais de la transition *PutPotential*[3]. Si les nouvelles valences polarisées ajoutées permettent de former des couples de valences duales dans le potentiel, satisfaisant le principe FA, alors les transitions *Dist-Left* et *Dist-Right* peuvent être appliquées. Puis, si le mot courant possède des dépendants locaux (dépendances projectives ou ancres) à gauche, la transition *Local-Left* est appliquée autant de fois qu'il y a de dépendants. Ces dépendances sont ajoutées de la plus proche du mot courant à la plus éloignée. Ensuite, si le mot courant est subordonné via une dépendance gauche ou gouverneur via une dépendance droite alors il devra être traité ultérieurement. La transition *Shift* est en ce cas appliquée. Si ce n'est pas le cas, alors le mot courant peut être traité immédiatement. Dans ce cas, si ce mot est subordonné via une dépendance droite alors la transition *Local-Right* est appliquée car il ne lui reste aucun

[2] La fonction qui pour une configuration donnée renvoie une transition est également appelé **oracle** (Nivre, 2008).
[3] L'algorithme de la fonction Valence utilisée dans l'algorithme 5 est présenté dans l'annexe D.

Algorithme 5 : CONVERSION

Données : $G = \{V,A\}$ une structure de dépendances pour la phrase $W = w_1...w_n$.

$\sigma \leftarrow [w_0]$
$\beta \leftarrow [w_1,\ldots,w_n]$
$\theta \leftarrow ()$
$k \leftarrow 1$
$A' \leftarrow \emptyset$
$c_0 = (\sigma,\beta,\theta,k,A')$
$fin \leftarrow$ faux
tant que $\neg fin$ **faire**
  $i \leftarrow$ INDICE($\beta[0]$)
  si $i = k$ **alors**
    $k = k + 1$
    $D \leftarrow \{d_1,\ldots,d_n\}$ **où** $\forall\ p \in \{1,\ldots,n\}$, $d_p = (i,l_p,t,j_p) \mid (j_p,l_p,t,i) \in A$ **avec** t =non-proj **et** $j_1 < \ldots < j_n$
    PutPotential(VALENCE(d_1,i),...,VALENCE(d_n,i))
    tant que $\exists$ *des paires* $\swarrow l\ \nwarrow l$ *ou* $\nearrow l\ \searrow l$ *dans* θ *satisfaisant le principe FA* **faire**
      si $\theta = \theta_1 \swarrow l\ \theta_2 \nwarrow l\ \theta_3$ **alors**
        Dist-Left(l)
      sinon si $\theta = \theta_1 \nearrow l\ \theta_2 \searrow l\ \theta_3$ **alors**
        Dist-Right(l)
  $D \leftarrow \{d_1,\ldots,d_n\}$ **où** $\forall\ p \in \{1,\ldots,n\}$, $d_p = (i,l_p,t,j_p) \in A$ **avec** t =projectif|ancre **et** $i > j_n > j_1$
  pour $k \leftarrow n$ **à** 1 **faire**
    Local-Left(l_k)
  si $\forall j > i\ \ \exists\ d = (j,l,t,i) \in A$ **ou** $\forall p > i\ \ \exists\ d = (i,l,t,p) \in A$ **alors**
    Shift
  sinon si $\forall j < i\ \exists\ d = (j,l,t,i) \in A$ **alors**
    Local-Right(l)
  sinon
    Shift
    $fin \leftarrow$ vrai

subordonné à rattacher. Pour finir, si aucune des conditions précédentes n'est remplie alors il s'agit de la racine w_0[4]. La transition *Shift* est appliquée et l'algorithme prend fin.

Un exemple de conversion pour la phrase donnée en exemple dans la figure 6.1 est exposé dans le tableau 6.2. Dans cet exemple, deux dépendances sont non-projectives. Les valences polarisées correspondantes sont rajoutées lors du premier traitement de chacun des mots impliqués dans ces dépendances, i.e. sont appliquées les transitions :

- *PutPotential*($\swarrow$coref) lors du premier traitement du mot « victoire » ;
- *PutPotential*($\nwarrow$coref$\swarrow$clit-a-obj) lors du premier traitement du mot « l' » (le nom de l'étiquette

[4]On considère ici que la structure à convertir est une structure complète, c'est à dire que chaque mot est dépendant via exactement une dépendance locale (dépendance projective ou ancre). La structure contenant uniquement les dépendances projectives et les ancres est une structure projective complète.

en dépendance est abrégé en *clit* dans l'exemple) ;

- *PutPotential*(↖clit-a-obj) lors du premier traitement du mot « méritée ».

Lorsque les couples ↙coref↖coref et ↙clit↖clit apparaissent dans le potentiel, la transition *Dist-Left* est appliquée. En conséquence les deux dépendances non-projectives correspondantes sont ajoutées à la structure, il s'agit d'une dépendance gauche étiquetée *coref* (coréférence) allant du mot « l' » au mot « victoire » et d'une dépendance gauche étiquetée *clit-a-obj* (clitique référant à un objet accusatif) de « méritée » à « l' ».

Transition	**Configuration**			
	([w_0],	[Cette,...,.],	(),	1, $\emptyset$)
Shift	⇒ ([...,Cette],	[victoire,...,.],	(),	2, $\emptyset$)
PutPotential	⇒ ([...,Cette],	[victoire,...,.],	(↙coref),	3, $\emptyset$)
Local-Left	⇒ ([w_0],	[victoire,...,.],	(↙coref),	3, $A = \{(2, \text{proj}, \text{det}, 1)\}$)
Shift	⇒ ([...,victoire],	[,,...,.],	(↙coref),	3, A)
Local-Right	⇒ ([w_0],	[victoire,...,.],	(↙coref),	4, $A_1 = A \cup \{(2, \text{proj}, \text{punct}, 3)\}$)
Shift	⇒ ([...,victoire],	[elle,...,.],	(↙coref),	4, A_1)
Shift	⇒ ([...,elle],	[l',...,.],	(↙coref),	5, A_1)
PutPotential	⇒ ([...,elle],	[l',...,.],	(↙coref↖coref↙clit),	6, A_1)
Dist-Left	⇒ ([...,elle],	[l',...,.],	(↙clit),	6, $A_2 = A_1 \cup \{(5, \text{non-proj}, \text{coref}, 2)\}$)
Shift	⇒ ([...,l'],	[a,...,.],	(↙clit),	6, A_2)
Local-Left	⇒ ([...,elle],	[a,...,.],	(↙clit),	6, $A_3 = A_2 \cup \{(6, \text{ancre}, \text{clit}, 5)\}$)
Local-Left	⇒ ([...,victoire],	[a,...,.],	(↙clit),	6, $A_4 = A_3 \cup \{(6, \text{proj}, \text{pred}, 4)\}$)
Local-Left	⇒ ([w_0],	[a,...,.],	(↙clit),	6, $A_5 = A_4 \cup \{(6, \text{ancre}, \text{coref}, 2)\}$)
Shift	⇒ ([...,a],	[méritée .],	(↙clit),	7, A_5)
PutPotential	⇒ ([...,a],	[méritée .],	(↙clit↖clit),	8, A_5)
Dist-Left	⇒ ([...,a],	[méritée .],	(),	8, $A_6 = A_5 \cup \{(7, \text{non-proj}, \text{clit}, 5)\}$)
Local-Right	⇒ ([w_0],	[a .],	(),	8, $A_7 = A_6 \cup \{(6, \text{proj}, \text{aux}, 7)\}$)
Shift	⇒ ([...,a],	[.],	(),	8, A_7)
Local-Right	⇒ ([w_0],	[a],	(),	9, $A_8 = A_7 \cup \{(6, \text{proj}, \text{punct}, 8)\}$)
Local-Right	⇒ ([],	[w_0],	(),	9, $A_9 = A_8 \cup \{(0, \text{proj}, \text{S}, 6)\}$)
Shift	⇒ ([w_0],	[],	(),	9, A_9)

TABLE 6.2 – Conversion de la structure de dépendances de la phrase « Cette victoire, elle l'a méritée. » en une séquence de transitions.

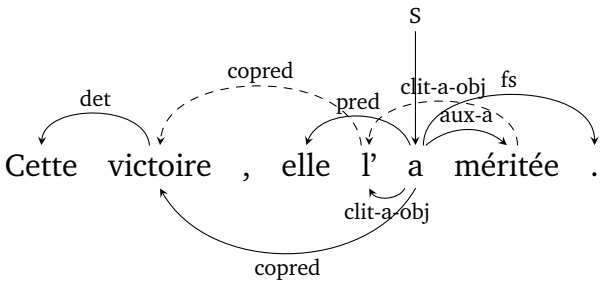


FIGURE 6.1 – Structure de dépendances pour la phrase « Cette victoire, elle l'a méritée. ».

6.3.2 Intégration d'un SVM

Pour qu'un système par transition soit efficace en terme de précision des analyses, il est nécessaire que son système de prédiction soit performant. Le système de prédiction doit vérifier les conditions

d'application des transitions et tenter de prédire la bonne transition en fonction des informations apportées par la configuration.

Dans le but d'optimiser la prédiction des transitions, nous avons choisi d'intégrer au système de prédiction un classificateur linéaire qui, à partir d'un apprentissage supervisé et des traits extraits des configurations, estime la transition la plus probable à appliquer à un temps donné de l'analyse (i.e. pour une configuration donnée). Parmi les classificateurs existants, notre choix s'est porté sur les SVM (*Support Vector Machine* ou séparateur à vaste marges, voir chapitre 3) pour leur efficacité déjà révélée dans divers travaux étudiant les systèmes d'analyse par transition (Yamada et Matsumoto, 2003; Cheng *et al.*, 2005a; Hall, 2006). Les SVM sont à l'origine prévus pour résoudre des problèmes de classification binaire (ou de régression) mais sont également adaptés à la classification multi-classes. Dans nos travaux nous avons besoin de résoudre un problème multi-classes où les classes sont les différentes transitions, associées à leurs paramètres (les étiquettes des dépendances et les valences polarisées), que l'on souhaite prédire. À partir de traits qui lui sont donnés en entrée, la fonction de prédiction du SVM calcule et renvoie la transition la plus probable (ou la liste des transitions dans l'ordre de leurs probabilités). Le système de prédiction calcule les probabilités des transitions en fonction des valeurs attribuées entre autres aux traits et aux couples vecteur de traits/classe. Ces valeurs, poids, sont calculés à partir d'une étape d'apprentissage sur des données correspondant à un ensemble de séquences de configuration/transition. Cet ensemble est obtenu par la conversion en séquences de transitions (détaillée dans la section précédente) d'un ensemble de structures de dépendances correctement annotées. Les traits, rassemblés sous forme de vecteur, correspondent à des informations extraites des configurations. Pour apprendre les poids il est donc nécessaire d'obtenir un ensemble de données correspondant à des couples configuration/transition (signifiant que pour une configuration particulière une transition particulière doit être appliquée). Chaque configuration est alors traduite en un vecteur de traits et chaque transition en une étiquette de classe.

Techniquement, les SVM travaillent avec des vecteurs de traits numériques et nous employons dans nos travaux une mise en œuvre des SVM qui gèrent les traits de la même façon. On ne peut donc pas spécifier directement, par exemple, l'étiquette grammaticale d'un mot. Pour une configuration ($[\ldots,w_n^\sigma]$,$[w_0^\beta,\ldots]$,$(v_1,\ldots,v_p)$,k,A), si l'on souhaite indiquer l'étiquette grammaticale E de w_0^β il faut définir un trait binaire indiquant par 1 si e(w_0^β)=E et par 0 sinon (pour une fonction e renvoyant l'étiquette grammaticale du mot en entrée). Un mot pouvant appartenir à une classe parmi 28 classes grammaticales (on considère ici le cas où les mot sont annotés suivant le jeu d'étiquettes précédemment employé dans la section 5.4.2), il est alors nécessaire de définir 28 traits binaires pour spécifier la classe du premier mot de la mémoire tampon β. Il est donc nécessaire de définir 28 autres traits pour chaque mot pour lequel nous souhaiterions indiquer l'étiquette grammaticale dans les traits. Outre l'exemple des étiquettes grammaticales, il peut être aussi nécessaire de spécifier directement les mots et dans ce cas, le nombre de traits binaires à définir dépend du nombre de mots compris dans le corpus d'apprentissage. Le nombre total de traits à prendre en compte lors de l'étape d'apprentissage peut donc être très élevé. Néanmoins, les SVM sont capables de gérer ce très grand nombre de traits et par ailleurs, il n'est pas nécessaire de spécifier dans le vecteur de traits correspondant à une configuration tous les traits non pertinents. Par exemple, pour un mot et un ensemble de traits d'indice 1 à 28 dénotant son appartenance à chacune des étiquettes grammaticales possibles, il est inutile de préciser dans le vecteur de traits les traits étant à 0. En ce cas, un vecteur de traits est une liste d'indices pertinents pour une configuration donnée. Les traits de base que nous avons choisis pour notre méthode d'analyse sont présentés dans le tableau 6.3.

Mots	Étiquettes grammaticales	Étiquettes de dépendances	Valences
w_n^σ, w_0^β,w_G	e_{n-2}^σ, e_{n-1}^σ, e_n^σ, e_0^β, e_1^β, e_2^β, e_G	lg_n^σ, ld_n^σ, lg_0^β, l_G	$v_0 \dots v_p$

TABLE 6.3 – Patrons de traits pour une configuration ([...,w_n^σ],[w_0^β,...],(v_0,...,v_p),k,A). ld_n^σ correspond à l'étiquette de la dépendance la plus à droite partant du mot (i.e. w_n^σ) du haut de la pile σ et les lg correspondent chacun à l'étiquette de la dépendance la plus à gauche partant des mots correspondants (i.e. w_n^σ et w_0^β). w_G, e_G et l_G correspondent respectivement au mot gouverneur du mot courant (s'il existe), à son étiquette grammaticale et à l'étiquette de la dépendance provenant de ce mot gouverneur.

6.4 Évolution du système

L'étude du formalisme présenté précédemment nous a amené à mettre en évidence différents problèmes et à y répondre dans la poursuite du développement de l'analyseur par transition. Nos travaux nous ont alors amené à proposer de nouveaux formalismes au fur et à mesure de l'évolution du système. Nous présentons dans cette section ces différentes évolutions.

6.4.1 Suppression du compteur

Le compteur apparaissant dans les configurations sert à limiter l'emploi de la transition *PutPotential*. Nous avons vu que la transition *PutPotential* ne pouvait être appliquée qu'une seule fois à un mot mais que celle-ci accepte un nombre indéfini de valences polarisées en paramètre qui permet l'ajout de plusieurs valences en même temps pour un mot. Dans l'exemple de conversion d'une structure de dépendances en séquence de transitions que nous avons exposé dans la sous-section 6.3.1, la transition *PutPotential(↖coref↙clit-a-obj)* est appliquée lors du traitement du mot « l' ». Une telle transition n'apparaît que rarement parmi les transitions appliquées sur un ensemble de données d'apprentissage et il est peu probable qu'une telle transition soit prédite lors de l'analyse des phrases tests car elle tente de prédire deux informations différentes en même temps. Il s'agit de la prédiction de deux dépendances non-projectives, l'une partant du mot (il s'agit d'une dépendance « coref » gauche) et l'autre arrivant (il s'agit d'une dépendance « clit-a-obj » gauche) sur le mot. Il semble donc plus intuitif de diviser cette transition en deux transitions distinctes (*PutPotential(↖coref)* et *PutPotential(↙clit-a-obj)*) qui ont une plus grande probabilité d'apparaître dans les données et d'être ensuite prédites lors de l'analyse.

En ce cas, nous proposons de modifier le formalisme de base pour permettre l'emploi de la transition *PutPotential* plusieurs fois sur un même mot. Pour ce faire, nous retirons l'usage du compteur « k » dans les configurations. Les configurations évoluent sous la forme de quadruplets $\langle \sigma, \beta, \theta, A \rangle$. Les transitions sont présentées dans le tableau 6.4. La transition *PutPotential* y est remplacée par la transition *PutValency* qui ne permet d'ajouter au potentiel qu'une seule valence polarisée à la fois. Les conditions, inhérentes au compteur, sont oubliées.

En outre, le compteur permettait également de s'assurer qu'un mot ne puisse recevoir qu'une seule dépendance non-projective. En supprimant le compteur, cette condition n'est plus certifiée puisqu'il est possible d'ajouter au potentiel plusieurs fois une valence de même orientation. Par exemple, si deux valences, de la forme $\swarrow l$ et $\swarrow l'$, sont associées à un même mot w_i alors il est possible qu'elles soient toutes les deux complétées ensuite par des valences de la forme $\nwarrow l$ et $\nwarrow l'$, créant deux dépendances non-projectives $(j,l,non\text{-}proj,i)$ et $(j',l',non\text{-}proj,i)$ arrivant sur ce même mot ($\forall j, j'$ tel que $w_i < w_j,w_{j'}$). Pour éviter ce genre de cas, nous ajoutons alors des conditions à la transition *PutValency* qui permettent d'une part d'interdire à une valence de la forme $\swarrow l$ d'être

Transition	**Effets sur les configurations**
Local-Left(l)	$(\sigma \mid w_i, w_j \mid \beta, \theta, A) \Rightarrow (\sigma, w_j \mid \beta, \theta, A \cup \{(j, l, proj \mid ancre, i)\})$
Local-Right(l)	$(\sigma \mid w_i, w_j \mid \beta, \theta, A) \Rightarrow (\sigma, w_i \mid \beta, \theta, A \cup \{(i, l, proj \mid ancre, j)\})$
Shift	$(\sigma, w_i \mid \beta, \theta, A) \Rightarrow (\sigma \mid w_i, \beta, \theta, A)$
PutValency(v)	$(\sigma, w_i \mid \beta, \theta, A) \Rightarrow (\sigma, w_i \mid \beta, \theta v^i, A)$
Dist-Left(l)	$(\sigma, \beta, \theta_1 \swarrow l^i\ \theta_2 \nwarrow l^j\ \theta_3, A) \Rightarrow (\sigma, \beta, \theta_1\theta_2\theta_3, A \cup \{(j, l, non\text{-}proj, i)\})$
Dist-Right(l)	$(\sigma, \beta, \theta_1 \nearrow l^i\ \theta_2 \searrow l^j\ \theta_3, A) \Rightarrow (\sigma, \beta, \theta_1\theta_2\theta_3, A \cup \{(i, l, non\text{-}proj, j)\})$
Conditions	
Local-Left	$i \neq 0$
PutValency($\swarrow l$)	$\forall l'$, $\swarrow l'^i \notin \theta$
PutValency($\searrow l$)	$\forall j, l'$, $(j,l',non\text{-}proj,i) \notin A$
Dist-Left	si $\swarrow l^i$ $\nwarrow l^j$ est la paire la plus à gauche satisfaisant la condition FA
Dist-Right	si $\nearrow l^i$ $\searrow l^j$ est la paire la plus à gauche satisfaisant la condition FA

TABLE 6.4 – Définition des transitions du système d'analyse par transition sans la prise en compte du compteur dans les configurations.

ajoutée au potentiel si une valence de cette même forme ($\forall l$) associée au mot courant existe déjà dans le potentiel et d'autre part d'interdire l'ajout d'une valence de la forme $\searrow l$ si le mot courant est déjà dépendant via une dépendance non-projective. À l'aide de ces conditions, un mot ne peut être dépendant que au plus d'une dépendance non-projective.

6.4.2 Adaptation des transitions locales à la méthode *arc-eager*

Une des erreurs fréquemment commises lors de l'analyse en dépendances par transition est le rattachement prématuré d'un dépendant droit. En effet, lorsqu'un mot obtient un gouverneur, celui-ci est retiré de σ (dans le cas d'une dépendance gauche) ou de β (dans le cas d'une dépendance droite). Un mot rattaché par une dépendance gauche a nécessairement été traité au moins une fois auparavant. Il a donc déjà eu l'occasion de rattacher ses dépendants gauches et droits. Tandis qu'un mot rattaché à droite n'a pas forcement encore vu tous ses dépendants droits. Il se peut alors qu'une dépendance droite soit placée trop tôt. Un exemple est donné par la figure 6.2. Dans cet exemple, la dépendance *a-obj* (objet accusatif) qui rattache « connaissances » à son gouverneur est assignée trop tôt car le mot « expérience » attend d'être rattaché à « connaissance » par une dépendance *n_aggr* (agrégation de noms).

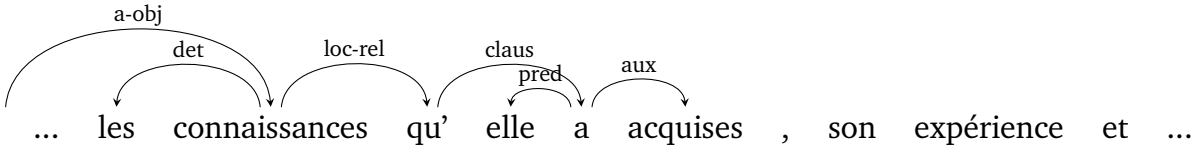


FIGURE 6.2 – Sous-structure de dépendances partielle extraite au cours d'une analyse par transition.

La méthode appelée *arc-eager* (Nivre, 2003) a été proposée pour résoudre ce genre de problèmes pour l'analyse en dépendances projective. Le principe de cette méthode est de rattacher les dépendants droits dès que possible. Cependant, les mot rattachés ainsi ne sont pas supprimés immédiatement car ils peuvent eux-même rattacher des dépendants droits ultérieurement. Par rapport à la méthode *arc-standard*, celle-ci inclut une transition supplémentaire et propose une

application différente de la transition *Right-Arc* (transition équivalente à *Local-Right*). La transition *Right-Arc* crée une dépendance entre le mot du haut de la pile σ et le premier mot de la mémoire tampon β. Cependant, le premier mot de β (le dépendant) n'est pas supprimé mais déplacé sur σ. Ainsi il peut rattacher d'autres mots à sa droite. Il est aussi possible qu'il n'ait pas de dépendant à rattacher. En ce cas, la transition *Reduce* permet de supprimer le mot du haut de la pile σ si celui-ci a déjà un gouverneur.

Dans le but d'améliorer les prédictions des dépendances projectives avec notre système d'analyse par transition, nous décidons d'y intégrer les principes de la méthode *arc-eager*. La méthode *Local-Right* est donc rectifiée afin de ne pas supprimer le dépendant lors de l'ajout d'une dépendance droite et la transition *Reduce* est rajoutée. L'ensemble des transitions et de leurs conditions d'application est présenté dans le tableau 6.5.

Transition	**Effets sur les configurations**
Local-Left(l)	$(\sigma \mid w_i, w_j \mid \beta, \theta, A) \Rightarrow (\sigma, w_j \mid \beta, \theta, A \cup \{(j, l, proj \mid ancre, i)\})$
Local-Right(l)	$(\sigma \mid w_i, w_j \mid \beta, \theta, A) \Rightarrow (\sigma w_i w_j, \beta, \theta, A \cup \{(i, l, proj \mid ancre, j)\})$
Reduce	$(\sigma \mid w_i, \beta, \theta, A) \Rightarrow (\sigma, \beta, \theta, A)$
Shift	$(\sigma, w_i \mid \beta, \theta, A) \Rightarrow (\sigma \mid w_i, \beta, \theta, A)$
PutValency(v)	$(\sigma, w_i \mid \beta, \theta, A) \Rightarrow (\sigma, w_i \mid \beta, \theta v^i, A)$
Dist-Left(l)	$(\sigma, \beta, \theta_1 \swarrow l^i\ \theta_2 \nwarrow l^j\ \theta_3, A) \Rightarrow (\sigma, \beta, \theta_1\theta_2\theta_3, A \cup \{(j, l, non\text{-}proj, i)\})$
Dist-Right(l)	$(\sigma, \beta, \theta_1 \nearrow l^i\ \theta_2 \searrow l^j\ \theta_3, A) \Rightarrow (\sigma, \beta, \theta_1\theta_2\theta_3, A \cup \{(i, l, non\text{-}proj, j)\})$
Conditions	
Local-Left	$i \neq 0 \wedge \neg\exists k \exists l' (k,l',i) \in A$
Local-Right	$\neg\exists k \exists l' (k,l',j) \in A$
Reduce	$\exists k \exists l (k,l,i) \in A$
PutValency($\swarrow l$)	$\forall l'$, $\swarrow l'^i \notin \theta$
PutValency($\searrow l$)	$\forall j, l'$, $(j,l',non\text{-}proj,i) \notin A$
Dist-Left	si $\swarrow l^i \nwarrow l^j$ est la paire la plus à gauche satisfaisant la condition FA
Dist-Right	si $\nearrow l^i \searrow l^j$ est la paire la plus à gauche satisfaisant la condition FA

TABLE 6.5 – Définition des transitions du système d'analyse par transition adapté aux principes de la méthode *arc-eager*.

6.4.3 Automatisation de l'ajout des dépendances non-projectives

Un constat notable est que les transitions *Dist-Left* et *Dist-Right* sont toujours employées immédiatement après l'application d'une transition *PutValency*. En pratique, elles sont employées dès lors qu'un couple de valences polarisées permettant de définir une dépendance non-projective apparaît dans le potentiel θ. Il en ressort clairement que ces transitions devraient non pas être prédites mais être automatiquement appliquées lorsqu'une valence polarisée permettant de compléter une dépendance non-projective est ajoutée au potentiel. Nous choisissons donc d'exclure les transitions *Dist-Left* et *Dist-Right* du système et d'intégrer leurs effets à la transition *PutValency*. Ainsi, lors de l'ajout d'une valence polarisée au potentiel θ, dès qu'une paire de valences polarisées, satisfaisant le principe FA, est trouvée, la dépendance non-projective correspondante est crée. Les différentes applications de la transition *PutValency* sont présentées dans le tableau 6.6.

Transition	**Effets sur les configurations**
PutValency($\swarrow l$)	$(\sigma, w_i \mid \beta, \theta, A) \Rightarrow (\sigma, w_i \mid \beta, \theta \swarrow l^i, A)$
PutValency($\nwarrow l$)	$(\sigma, w_i \mid \beta, \theta_1 \swarrow l^j \theta_2, A) \Rightarrow (\sigma, w_i \mid \beta, \theta_1\theta_2, A \cup \{(i, l, non\text{-}proj, j)\})$
PutValency($\nearrow l$)	$(\sigma, w_i \mid \beta, \theta, A) \Rightarrow (\sigma, w_i \mid \beta, \theta \nearrow l^i, A)$
PutValency($\searrow l$)	$(\sigma, w_i \mid \beta, \theta_1 \nearrow l^j \theta_2, A) \Rightarrow (\sigma, w_i \mid \beta, \theta_1\theta_2, A \cup \{(j, l, non\text{-}proj, i)\})$
Conditions	
PutValency($\swarrow l$)	$\forall l'$, $\swarrow l'^i \notin \theta$
PutValency($\nwarrow l$)	$\forall k$, $\swarrow l^k \notin \theta_2$
PutValency($\searrow l$)	$\forall j, l'$, $(j,l',non\text{-}proj,i) \notin A \ \wedge \ \forall k \nearrow l^k \notin \theta_2$

TABLE 6.6 – Effet de l'application (et conditions) de la transition *PutValency* sur les configurations en fonction de la polarité de la valence passée en paramètre à la transition. La condition associé à la transition PutValency($\nwarrow l$) permet de vérifier le principe **FA**.

6.4.4 Séparation des modèles de prédiction

Le nombre d'étiquettes de dépendances étant relativement grand, on y trouve une part importante d'étiquettes peu fréquentes dans les données d'apprentissage. Ces étiquettes représentent généralement des fonctions syntaxiques très précises figurant dans des contextes particuliers. Par ailleurs, les dépendances non-projectives étant également moins fréquentes que les dépendances projectives, elles sont plus difficiles à prédire et les étiquettes qui leurs sont associées sont encore plus rares dans les données.

En divisant l'étape de prédiction des étiquettes en deux étapes successives, de prédiction des transitions d'une part et de prédiction des paramètres d'autres part, nous espérons obtenir des modèles mieux adaptés à chaque transition. Il s'agit donc, dans un premier temps, d'apprendre un premier modèle général permettant de prédire les noms des transitions, i.e. *Local-Left*, *Local-Right*, *Shift*, *Reduce* et *PutValency*. Dans un second temps, un modèle supplémentaire est appris pour chaque transition nécessitant des paramètres. De ce fait, trois modèles supplémentaires sont appris :

- un modèle pour la transition *Local-Left* permettant de prédire l'étiquette syntaxique d'une dépendance gauche ;
- un modèle pour la transition *Local-Right* permettant de prédire l'étiquette syntaxique d'une dépendance droite ;
- un modèle pour la transition *PutValency* permettant de prédire une valence polarisée.

En pratique, cela se traduit par une réduction du nombre de classes à prédire pour chaque modèle. Par exemple, l'apprentissage d'un modèle unique sur le CDG Treebank révèle 258 classes (i.e. des couples transition/paramètre) avec lesquelles le SVM doit travailler tandis que la séparation des modèles produit un premier modèle gérant uniquement 5 classes (les transitions), puis trois autres modèles maniant respectivement 77 classes pour les dépendances gauches, 109 pour les dépendances droites et 70 pour les valences polarisées.

6.5 Expérimentations, résultats et discussion

Dans le but d'évaluer les performances du système d'analyse en dépendances par transition et le bénéfice apporté par chacune des évolutions que nous proposons, nous expérimentons sur des données annotées selon le schéma d'annotation des grammaires catégorielles de dépendances.

• **L'ensemble de données** que nous utilisons pour effectuer les expérimentations est le CDG Treebank (voir la sous-section 4.4.2 du chapitre 4). Nous conservons pour les expérimentations la bonne segmentation des phrases en mots et apprenons les modèles sur des données dont l'étiquetage grammaticale résulte de la conversion des classes grammaticales en un jeu d'étiquettes standard (voir la sous-section 5.4.2 du chapitre 5).

• **Les expérimentations** sont effectuées pour chaque évolution du système. Chaque stade de l'évolution du système inclut les évolutions précédentes. Pour effectuer les expérimentations, chaque ensemble de test (les phrases correctement segmentées) est d'abord pré-étiqueté grammaticalement par Melt (Denis et Sagot, 2009, 2012) puis analysé selon les différents stade d'évolution du système par transition :

- formalisme de base (0) ;
- suppression du compteur (1) ;
- intégration de la méthode *arc-eager* (2) ;
- automatisation de l'ajout des dépendances projectives et séparation des modèles de prédiction (3).

Nous réalisons des évaluations croisées sur les données (10 strates). Les évaluations comprennent l'estimation des scores d'attachement et de précision sur les mots. Nous rappelons les critères d'évaluation :

- le LA (*Label Accuracy*) est le pourcentage de mots pour lesquels le bon nom de la dépendance arrivant sur ce mot a été assigné ;
- le LAS (*Labelled Attachement Score*) est le pourcentage de mots pour lesquels la bonne dépendance et la bonne étiquette de dépendance ont été assignées ;
- le UAS (*Unlabelled Attachement Score*) est le pourcentage de mots pour lesquels la bonne dépendance (sans tenir compte de son nom) a été assignée.

Par ailleurs, le système étant dédié à l'analyse en dépendances non-projective, nous choisissons d'estimer les scores sur les mots associés à des dépendances projectives (dans la structure de dépendances correctement annotée du corpus) d'une part et les scores sur les mots associés à des dépendances non-projectives d'autre part. Les scores sur les dépendances projectives ne prennent pas en compte les ancres (qui sont des dépendances outils) ni les dépendances associées à des ponctuations.

• **Les résultats** sont présentés dans le tableau 6.7. Les premiers scores obtenus par l'analyseur exploitant le système de base sont intéressants mais restent largement en dessous de scores obtenus avec des méthodes communément employées dans le domaine de l'analyse en dépendances par transition (e.g. $\approx$ 5,5 points de plus en LAS sur les mêmes données pour les méthodes non-projectives et pseudo-projectives du MaltParser telles que *covnonproj* et *arc-eager*). Par ailleurs, les scores sur les dépendances non-projectives (de 45,6 % à 51,5 % en LAS) sont beaucoup plus bas que les scores sur les dépendances projectives (de 73,8 % à 75,3 % en LAS). Seulement une dépendance non-projective sur deux est correctement assignée. Les plus hauts scores globaux sont obtenus avec la mise en œuvre de la seconde évolution (2), le score en LAS est de +1,6 par rapport au formalisme de base.

	Toutes dépendances			Dép. Projectives			Dép. Non-projectives		
	LA	**UAS**	**LAS**	**LA**	**UAS**	**LAS**	**LA**	**UAS**	**LAS**
(0) Formalisme de base	75,3	82,4	72,6	76,3	83,8	73,8	51,8	51,8	45,6
(1) Suppression du compteur	76,0	82,8	73,2	76,8	83,9	74,1	**59,1**	**57,9**	**51,5**
(2) Intégration de *arc-eager*	**77,2**	83,9	**74,2**	**77,9**	85,2	**75,3**	57,0	54,2	47,9
(3) Autom. & Séparation	76,3	**84,1**	73,5	77,2	**85,4**	74,7	57,6	55,1	47,7

TABLE 6.7 – Comparaison des résultats d'analyse par transition selon l'évolution du système.

Notons que la première évolution du système (1) permet d'augmenter fortement les scores sur les dépendances non-projectives (+5,9 LAS). La suppression du compteur dans les configurations permet une meilleure prédiction des valences polarisées. Les scores sur les dépendances projectives sont également légèrement supérieurs (+0,3 LAS), ce qui globalement occasionne de meilleurs scores sur l'ensemble des dépendances (+0,6 LAS).

Suite à cette amélioration, l'intégration des principes de la méthode *arc-eager* (2) dans le formalisme permet également d'obtenir des scores plus élevés. Cette méthode, ayant pour objectif d'améliorer l'attachement des dépendances à droite, permet en conséquence d'accroître les scores significativement sur les dépendances projectives (+1,2 LAS). Cependant, cette seconde évolution impacte négativement les scores sur les dépendances non-projectives, qui baissent de 3,6 points en LAS. Globalement, les dépendances projectives étant considérablement plus nombreuses que les dépendances non-projectives, les scores sont dans l'ensemble plus intéressants (+1,0 LAS).

La troisième évolution du système n'a pas ou peu d'impact sur les scores puisqu'elle consiste à automatiser l'ajout des dépendances non-projectives, celles-ci étant déjà prédites systématiquement après l'emploi d'une transition *PutValency*. Cette évolution est donc combinée à la dernière évolution du système (3). Cette dernière évolution, la séparation des modèles de prédiction, consiste d'une certaine manière à prédire les dépendances avant de prédire leurs étiquettes. On constate alors que les scores UAS sont légèrement (mais pas significativement) supérieurs (+ 0,2) aux scores précédents. Cependant, les scores prenant en compte la validité des étiquettes (LA et LAS) sont généralement inférieurs, excepté pour les dépendances non-projectives pour lesquelles le UAS est nettement supérieur (+ 0,9). Globalement, cette dernière évolution n'est pas suffisamment intéressante dans le cadre de notre méthode d'analyse par transition.

En outre, les ancres ne sont pas prises en compte dans le calcul des dépendances projectives. Cependant, calculer indépendamment les scores sur les ancres montre que celles-ci sont mieux prédites que les dépendances non-projectives auxquelles elles sont associées. En effet, les ancres obtiennent en moyenne 7 points de plus en LAS que les dépendances non-projectives et 10 points de plus en UAS. Ces scores sont donc nettement supérieurs aux scores obtenus sur les dépendances non-projectives mais également toujours très inférieurs aux scores obtenus sur les dépendances projectives. Bien que les ancres soient des dépendances projectives, elles n'obtiennent pas des résultats équivalents. Nous supposons que la mauvaise prédiction des ancres vient d'une part de leur faible fréquence dans les données[5] et d'autre part du fait du système de prédiction qui prédit en même temps le statut de la dépendance (ancre ou dépendance projective) et son étiquette.

Par ailleurs, un examen plus précis des scores obtenus sur les dépendances non-projectives révèle la forte disparité des scores entre les dépendances gauches et droites. Le tableau 6.8 présente ces résultats. On y voit clairement que les scores sur les dépendances non-projectives gauches

[5]La fréquence des ancres dans le CDG Treebank est identique à la fréquence des dépendances non-projectives (i.e. 3.8 %) puisque chaque dépendance non-projective est couplée à une ancre.

	Dép. Non-projectives			% parmi les
	LA	**UAS**	**LAS**	**dép. non-projectives**
Dépendances gauches	70,3	67,6	59,9	74,4
Dépendances droites	18,1	15,2	13,0	25,6

TABLE 6.8 – Résultats des dépendances non-projectives gauches et droites.

(59,9 LAS) sont considérablement meilleurs que les scores sur les dépendances non-projectives droites (13,0 LAS). En outre, nous observons que le nombre de dépendances droites représente seulement un quart de l'ensemble des dépendances non-projectives. Les faibles scores des dépendances droites peuvent donc être expliqués d'une part par leur faible présence dans l'ensemble de données. Néanmoins, nous supposons également que la méthode de prédiction des dépendances non-projectives n'est pas adaptée à la prédiction des dépendances droites. En effet, pour trouver une dépendance droite il est nécessaire de prédire une valence polarisée de la forme $\nearrow l$ lors du traitement du gouverneur de la dépendance avant de prédire la valence polarisée duale $\searrow l$ lors du traitement du dépendant. Il s'agit donc de trouver le gouverneur avant le dépendant, ce qui semble intuitivement très difficile à réaliser automatiquement sans informations supplémentaires.

6.6 Conclusion

Le système d'analyse par transition proposé et étudié dans ce chapitre est adapté à la représentation en dépendances induite par les grammaires catégorielles de dépendances. Ce système comprend des opérations permettant la gestion des dépendances projectives et des ancres d'une part, ainsi que des opérations permettant la gestion des dépendances non-projectives d'autre part inspirées de la gestion des dépendances non-projectives dans les CDG. Le système permet d'effectuer des analyses en dépendances par transition en un seul traitement de complexité linéaire. Dans le cadre de cette thèse, nous avons proposé tout d'abord l'intégration d'un outil de classification, un SVM, pour améliorer la prédiction des transitions et ainsi obtenir des premiers résultats intéressants.

Nous avons également proposé plusieurs évolutions du système répondant à différents problèmes inférés par le système de base. Les premières évolutions permettent d'améliorer significativement les scores d'analyse en dépendances tandis que la dernière évolution, dissociant la prédiction des dépendances de la prédiction des étiquettes, n'apporte pas d'amélioration globale. La suppression du compteur dans les configurations et l'intégration des principes de la méthode arc-eager sont deux évolutions pertinentes qui permettent d'améliorer respectivement les scores sur les dépendances non-projectives et les scores sur les dépendances projectives.

Néanmoins, bien que le système soit adapté à la représentation des CDG les scores que nous obtenons n'atteignent pas les scores que peuvent obtenir un analyseur standard tel que le MaltParser. En outre, le mécanisme proposé ne tire pas profit de la représentation mixte projective/non-projective des CDG. Les scores sur les dépendances non-projectives sont relativement bas. De plus, les scores sur les ancres sont légèrement plus élevés, mais également notablement plus bas que les scores sur les dépendances projectives, bien qu'elles soient elles-même des dépendances projectives. Il semble donc que le système actuel ne soit pas bien adapté à la prédiction simultané des dépendances projectives et non-projectives, et en particulier à la prédiction des dépendances non-projectives droites. Cependant, les évaluations, dont l'analyse approfondie des ancres et des dépendances non-projectives, ont mis en évidence certains problèmes, amenant à de nouvelles pistes dans le domaine de l'analyse en dépendances par transition.

7

Séparation des étapes de prédiction des dépendances projectives et non-projectives

7.1 Introduction

Les travaux que nous présentons dans ce chapitre font suite aux travaux exposés dans le chapitre précédent. Nous avions investigué une méthode d'analyse par transition et ses différentes évolutions dans le but de répondre à un problème particulier : globalement les méthodes d'analyse en dépendances par transition révèlent des scores bien moindres sur les dépendances non-projectives que sur les dépendances projectives. Nous proposions, dans le chapitre précédent, un système par transition inspiré du formalisme des grammaires catégorielles de dépendances gérant deux niveaux de représentation en un seul, i.e. combinant l'usage des dépendances projectives et non-projectives dans une même représentation. L'étude approfondie des résultats nous a permis de mettre en évidence les problèmes induits par ce système :

- la prédiction simultanée des dépendances projectives et non-projectives ne semble pas être efficace dans le cadre de la méthode proposée, les scores sur les dépendances non-projectives sont relativement bas. En outre, les scores globaux n'atteignent pas les scores que l'on peut obtenir avec des méthodes standards existantes ;
- la prédiction des dépendances non-projectives droites échoue très souvent alors que les dépendances non-projectives gauches obtiennent des scores très supérieurs.

Dans le but de résoudre les problèmes soulevés par la méthode proposée précédemment, nous présentons dans ce chapitre une méthode séparant la prédiction des dépendances projectives des dépendances non-projectives d'une part et séparant ensuite la prédiction des dépendances non-projectives gauches des dépendances non-projectives droites. Il s'agit donc dans un premier temps d'effectuer une analyse en dépendances projective afin de prédire les dépendances projectives et les ancres sans distinction des unes par rapport aux autres. Puis, nous présentons deux méthodes inverses permettant de prédire les dépendances non-projectives gauches d'une part et droites d'autre part.

Les corpus en dépendances employés pour l'entraînement des différentes étapes d'analyse contiennent alors nécessairement des structures de dépendances en adéquation avec la représentation induite par les CDG. Cependant, bien que les ancres soient des éléments découlant du formalisme des grammaires catégorielles de dépendances, il est possible de leur substituer des ancres artificielles générées à partir de structures de dépendances standards à l'aide d'une méthode de projectivisation. Nous comparons donc, dans un premier temps, les résultats de notre méthode obtenus sur le CDG Treebank d'origine avec les résultats obtenus sur ce même corpus mais dont les ancres d'origine ont été remplacées par des ancres produites automatiquement. Nous montrons alors, dans un second temps, que cette méthode est adaptable à d'autres langues et d'autres corpus en dépendances ne comprenant pas la notion d'ancre, et expérimentons sur ces corpus.

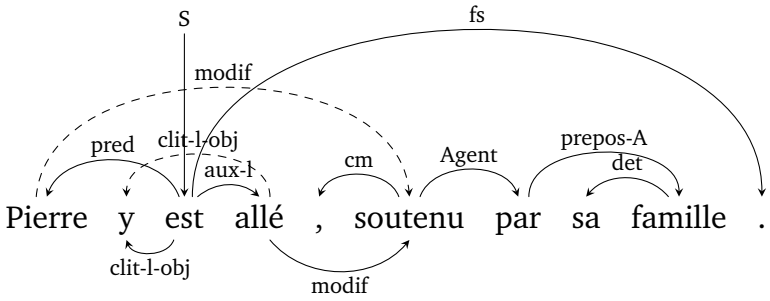


FIGURE 7.1 – Structure de dépendances, suivant la représentation induite par les CDG, pour la phrase « Pierre y est allé, soutenu par sa famille. ».

À travers la séparation des étapes de prédiction des dépendances projectives et non-projectives nous cherchons à tirer partie de l'efficacité de méthodes d'analyse en dépendances projectives pour améliorer ultérieurement la prédiction des dépendances non-projectives en utilisant des dépendances projectives « factices » secondant les dépendances non-projectives. En ce sens, notre méthode peut être comparée à la méthode pseudo-projective de Nivre et Nilsson (2005). Dans un premier temps, ils projectivisent automatiquement les arbres de dépendances non-projectifs en substituant des dépendances projectives aux dépendances non-projectives (et en ajoutant potentiellement des informations aux étiquettes de ces nouvelles dépendances projectives ; voir la section 2.3.2 du chapitre 2). Ils emploient ensuite une méthode d'analyse projective pour prédire l'ensemble des dépendances projectives (d'origine et projectivisées), puis utilisent un algorithme de déprojectivisation qui tente de retrouver mécaniquement les dépendances non-projectives à partir des informations fournies par les étiquettes des dépendances projectives. Notre méthode diffère de celle-ci par le fait que les étapes non-projectives que nous proposons exploitent le potentiel de méthodes d'apprentissage. Il ne s'agit pas de retrouver mécaniquement ces dépendances mais de prédire les couples de mots qui ont le plus de probabilité d'être liés par une dépendance non-projective. Nos travaux peuvent également être rapprochés de ceux de McDonald et Pereira (2006), pour l'analyse en dépendances basée sur les graphes, dans lesquels les dépendances sont réordonnées après l'analyse projective en se basant sur les poids des dépendances. En outre, la prédiction des dépendances non-projectives correspond à une étape de post-traitement telle que celle utilisée pour retrouver ces dépendances après la conversion d'un corpus des constituants aux dépendances (car la conversion induit généralement la projectivité des arbres obtenus). On peut trouver un exemple d'utilisation d'un tel post-traitement dans les travaux de Hall et Novák (2005).

7.2 Une méthode d'analyse en trois étapes

Nous proposons une méthode d'analyse en dépendances par transition comprenant trois étapes. Chaque étape est dédiée à une tâche particulière. Les systèmes par transitions employés pour chacune de ces étapes sont donc également spécifiques. Nous présentons ci-après chacun de ces systèmes.

7.2.1 Étape projective

L'étape projective étant une étape standard d'analyse en dépendances projective, nous décidons d'employer une méthode standard : la méthode *arc-eager* (Nivre, 2003). Notre choix s'est porté sur cette méthode car il s'agit de celle permettant d'obtenir les meilleurs résultats sur nos données, cependant il est bien sûr possible d'employer n'importe quelle méthode d'analyse en dépendances projective. Dans le chapitre précédent, nous avions intégré la méthode *arc-eager* à notre système pour améliorer la prédiction des dépendances projectives. Dans le cadre de ces travaux, nous utiliserons le MaltParser (Nivre *et al.*, 2006a) pour effectuer l'étape de prédiction des dépendances projectives puisque la méthode y est incluse.

Nous rappelons donc ici les éléments constituant le système d'analyse par transition *arc-eager*. Une configuration y est représentée par un triplet $\langle\sigma,\beta,A\rangle$ où :

- σ est une pile ;
- β est une mémoire tampon ;
- A est un ensemble d'arcs.

Les arcs (dépendances) de l'ensemble A sont ici représentés par des triplets de la forme (j,l,i) où j est l'indice du gouverneur, i est l'indice du dépendant et l est l'étiquette de la dépendance. Cette description des dépendances ne comprend pas d'information sur la sorte de l'arc [1] car nous considérons dans cette étape uniquement des dépendances projectives (sans distinction entre les dépendances projectives et les ancres).

Les transitions utilisées par ce système sont rappelées dans le tableau 7.1. Comme dans le chapitre précédent (sous-section 6.4.2), les transitions standards *Right-Arc* et *Left-Arc* sont renommées respectivement *Local-Right* et *Local-Left*. Elles permettent d'ajouter des dépendances droites et gauches si les mots dépendants ne sont pas déjà rattachés par d'autres dépendances. Pour chaque dépendance gauche ajoutée, le mot du haut de la pile σ (il s'agit du subordonné de la dépendance : w_i dans le tableau 7.1 lors de l'application de la transition *Local-Left*) est supprimé. Tandis que pour chaque dépendance droite ajoutée, le premier mot de β (w_j dans le tableau 7.1 lors de l'application de la transition *Local-Right*) est déplacé sur la pile σ. Le mot n'est pas supprimé lors de l'application de la transition *Local-Right* car il peut encore avoir des dépendants à droite. La transition *Reduce* permet de supprimer le mot du haut de la pile si celui-ci est rattaché par une dépendance et *Shift* fait passer le premier mot de β sur σ.

Pour une phrase $W = w_1 \dots w_n$ donnée, la configuration initiale d'une analyse est de la forme $([w_0],[w_1,\dots,w_n],\emptyset)$ où w_0 est la racine artificielle de la structure. À la fin d'une analyse, toute configuration finale est de la forme $(\sigma',[],A')$ où σ' est de la forme $[w_0,...]$ et A' contient un ensemble d'arcs correspondant à une structure de dépendances projective (partielle) pour la phrase

[1] Nous indiquions précédemment (voir la sous-section 6.2.1 du chapitre 6) s'il s'agissait d'une dépendance *projective*, d'une dépendance *non-projective* ou d'une *ancre*.

Transition	**Effets sur les configurations**	**Conditions**
Local-Left(l)	$(\sigma \mid w_i,w_j \mid \beta, A) \Rightarrow (\sigma,w_j \mid \beta,A \cup \{(j,l,i)\})$	$i \neq 0 \wedge \neg\exists k \exists l'(k,l',i) \in A$
Local-Right(l)	$(\sigma \mid w_i,w_j \mid \beta, A) \Rightarrow (\sigma \mid w_i w_j,\beta,A \cup \{(i,l,j)\})$	$\neg\exists k \exists l'(k,l',j) \in A$
Reduce	$(\sigma \mid w_i,\beta, A) \Rightarrow (\sigma,\beta,A)$	$\exists k \exists l(k,l,i) \in A$
Shift	$(\sigma,w_i \mid \beta, A) \Rightarrow (\sigma \mid w_i,\beta,A)$	

TABLE 7.1 – Définition des transitions du système *arc-eager*.

W. Les mots qui pourraient ne pas avoir été rattachés durant l'analyse (i.e. restant dans σ') sont automatiquement rattachés à la racine w_0 à la fin de l'analyse.

La structure projective correspondant à la structure non-projective de la figure 7.1 pour la phrase « Pierre y est allé, soutenu par sa famille » est présentée par la figure 7.2. Les dépendances non-projectives y sont exclues et les dépendances ancres sont considérées comme des dépendances projectives. La séquence de transitions permettant d'obtenir cette structure par la méthode d'analyse par transition *arc-eager* est donnée dans le tableau 7.2.

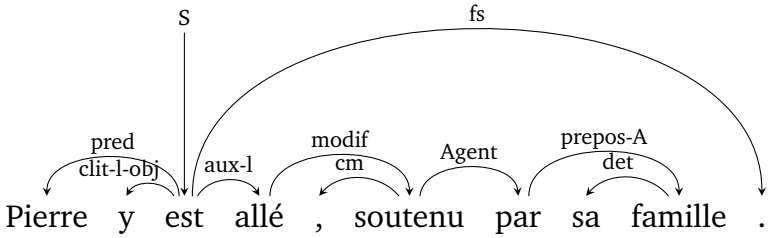


FIGURE 7.2 – Structure de dépendances projective pour la phrase « Pierre y est allé, soutenu par sa famille. ». Les dépendances étiquetées *clit-l-obj* et *modif* correspondent aux ancres de la structure non-projective (figure 7.1) pour cette même phrase. Dans cette structure elle sont considérées comme des dépendances projectives.

7.2.2 Étapes non-projectives gauche et droite

Le but des étapes non-projectives est de prédire les dépendances non-projectives gauches et droites indépendamment en tenant compte des informations apportées par la structure de dépendances projective produite durant l'étape préalable. Pour effectuer ces étapes nous proposons deux méthodes inverses inspirées des travaux présentés dans le chapitre précédent et donc de la gestion des valences dans les grammaires catégorielles de dépendances. Le principe de ces deux étapes est de prédire les paires de valences polarisées correspondant à des dépendances non-projectives en parcourant la phrase d'une part de gauche à droite et d'autre part de droite à gauche. Les configurations de ces deux systèmes sont des quadruplets $\langle \sigma, \beta, \theta, A \rangle$ où θ est une liste pouvant contenir des valences polarisées.

La première étape non-projective est l'étape de prédiction des dépendances gauches[2]. Dans cette étape les mots de la phrase sont parcourus dans leur ordre naturel. Chaque traitement d'un

[2]L'ordre dans lequel sont exécutées les étapes non-projectives n'a pas une importance majeure. Nous choisissons de présenter l'étape de prédiction des dépendances gauches en premier car elle semble plus naturelle à appréhender du fait de la lecture de la phrase de gauche à droite tandis que l'étape de prédiction des dépendances droites agit comme un miroir de l'étape de prédiction des dépendances gauches.

Transition		**Configuration**		
		([w_0],	[Pierre,...,.],	$\emptyset$)
Shift	$\Rightarrow$	([...,Pierre],	[y,...],	$\emptyset$)
Shift	$\Rightarrow$	([...,y],	[est,...],	$\emptyset$)
Local-Left(clit-l-obj)	$\Rightarrow$	([...,Pierre],	[est,...],	$A = \{(3,\text{clit-l-obj},2)\}$)
Local-Left(pred)	$\Rightarrow$	([w_0],	[est,...],	$A_1 = A \cup \{(3,\text{clit-l-obj},1)\}$)
Local-Right(S)	$\Rightarrow$	([...,est],	[allé,...],	$A_2 = A_1 \cup \{(0,\text{S},3)\}$)
Local-Right(aux-l)	$\Rightarrow$	([...,allé],	[,,...],	$A_3 = A_2 \cup \{(3,\text{aux-l},4)\}$)
Shift	$\Rightarrow$	([...,,],	[soutenu,...],	A_3)
Local-Left(cm)	$\Rightarrow$	([...,allé],	[soutenu,...],	$A_4 = A_3 \cup \{(6,\text{cm},5)\}$)
Local-Right(modif)	$\Rightarrow$	([...,soutenu],	[par,...],	$A_5 = A_4 \cup \{(4,\text{modif},6)\}$)
Local-Right(Agent)	$\Rightarrow$	([...,par],	[sa,...],	$A_6 = A_5 \cup \{(6,\text{Agent},7)\}$)
Shift	$\Rightarrow$	([...,sa],	[famille,...],	A_6)
Local-Left(det)	$\Rightarrow$	([...,par],	[famille,...],	$A_7 = A_6 \cup \{(9,\text{det},8)\}$)
Local-Right(prepos-A)	$\Rightarrow$	([...,famille],	[.],	$A_8 = A_7 \cup \{(7,\text{prepos-A},9)\}$)
Reduce	$\Rightarrow$	([...,par],	[.],	A_8)
Reduce **(x3)**	$\Rightarrow$	([...,est],	[.],	A_8)
Local-Right(fs)	$\Rightarrow$	([...,.],	[],	$A_9 = A_8 \cup \{(3,\text{fs},10)\}$)

TABLE 7.2 – Séquence de transitions permettant d'obtenir la structure de dépendances projective de la phrase « Pierre y est allé, soutenu par sa famille. ».

mot consiste à prédire si oui ou non une dépendance non-projective gauche arrive ou part de ce mot. Il s'agit donc de prédire les valences $\swarrow l$ (fin d'une dépendance gauche étiquetée l) ou $\nwarrow l$ (départ d'une dépendance gauche étiquetée l) pour chaque mot impliqué dans une dépendance non-projective gauche. Pour un couple de valence $\swarrow l \nwarrow l$ correspondant à une dépendance non-projective gauche, la valence $\swarrow l$ est donc nécessairement prédite avant sa valence duale $\nwarrow l$ dans la séquence de transitions. De plus, de part la définition des transitions, seules les valences de la forme $\swarrow l$ sont mémorisées dans la liste θ. Les transitions sont présentées dans le tableau 7.3.

Transition	**Effets sur les transitions**	**Conditions**
PutValency($\swarrow l$)	$(\sigma,w_i \mid \beta,\theta,A) \Rightarrow (\sigma \mid w_i,\beta,\theta\swarrow l^i,A)$	$\forall l' \exists! k \;\; (k,l',i) \in A$
Dist-Left($\nwarrow l$)	$(\sigma,w_j \mid \beta,\theta_1\swarrow l^i\theta_2,A) \Rightarrow (\sigma,w_j \mid \beta,\theta_1\theta_2,A \cup \{(j,l,i)\})$	$\swarrow l \notin \theta_2$
Shift	$(\sigma,w_i \mid \beta,\theta,A) \Rightarrow (\sigma \mid w_i,\beta,\theta,A)$	

TABLE 7.3 – Définition des transitions du système de prédiction des dépendances non-projectives gauches.

La transition *Shift* apparaissant dans ce système est standard, elle permet de déplacer le premier mot de β sur la pile σ. La transition *PutValency* permet de prédire, pour le premier mot de β, exactement une valence polarisée de la forme $\swarrow l$ signifiant qu'une dépendance non-projective gauche doit arriver sur ce mot. La valence est alors ajoutée à la fin de la liste θ et le mot (ici w_i) est déplacé sur la pile σ car il ne peut recevoir qu'une seule dépendance non-projective. Pour finir, la transition *Dist-Left* est appliquée lorsque le premier mot de β reçoit une valence polarisée de la forme $\nwarrow l$, c'est à dire quand une dépendance non-projective gauche doit partir de ce mot. La valence n'est pas ajoutée à θ mais sa valence duale la plus proche (i.e. la valence $\swarrow l \in \theta$ permettant de satisfaire le principe FA[3] pour le couple $\swarrow l \nwarrow l$) est supprimée de θ et la dépendance non-projective gauche étiquetée l correspondant au couple de valences est ajoutée à l'ensemble A.

[3]Voir la sous-section 4.3.3. Principe FA : les valences duales les plus proches forment un couple.

Lorsque cette transition est appliquée, le premier mot de β n'est pas déplacé sur σ car il peut être l'origine d'autres dépendances non-projectives gauches.

Pour une phrase $W = w_1,\ldots,w_n$ donnée, la configuration initiale d'une analyse avec ce système est $([w_0],[w_1,\ldots,w_n],(),A')$ où A' est l'ensemble des arcs construits lors de l'étape précédente de prédiction des dépendances projectives. Une configuration finale sera alors de la forme $([w_0,\ldots,w_n],[],\theta',A'')$ quelque soit θ' pouvant contenir des valences de polarité sud-ouest n'ayant pas trouvé leur valence duale et un ensemble d'arc A'' contenant des dépendances projectives et potentiellement non-projectives gauches.

La seconde étape non-projective peut-être considérée comme une méthode « inverse » de la précédente. Elle permet de prédire les dépendances droites en parcourant la phrase de droite à gauche. Les transitions du système d'analyse pour cette étape sont présentées dans le tableau 7.4. La configuration initiale d'une analyse avec un tel système est : $([w_0,w_1,\ldots,w_{n-1}],[w_n],(),A'')$ où A'' est l'ensemble d'arcs résultant de l'étape précédente. Une configuration finale pour ce système sera de la forme $([w_0],[w_1,\ldots,w_n],\theta'',A''')$ avec A''' représentant la structure de dépendances non-projective complète pour la phrase W.

La transition *PutValency* proposée pour ce système permet de prédire uniquement des valences de polarité sud-est (i.e. de la forme $\searrow l$). Le mot considéré par cette transition et recevant la valence est le mot du haut de la pile σ et non le premier mot de la mémoire tampon β. Alors, à chaque application de cette transition, la valence polarisée prédite est ajoutée au début de la liste θ et le mot (w_i) du haut de la pile σ est déplacée dans β. De même que l'application de la transition *PutValency* est inversée dans ce système par rapport au précédent, la transition *Dist-Right* correspond à l'inverse de la transition *Dist-Left*. L'application de la transition *Dist-Right* permet de compléter les paires de valences polarisées représentant des dépendances non-projectives droites par la prédiction de valences polarisées de la forme $\nearrow l$. Lors de l'emploi de cette transition, la valence duale à $\nearrow l$, c'est à dire $\searrow l$, la plus proche dans θ est supprimée. Les valences étant ajoutées dans la liste par la gauche, la valence la plus proche signifie la valence la plus à gauche dans la liste. Enfin, la transition *RShift* permet de déplacer le mot du haut de la pile σ dans β.

Transition	**Effets sur les transitions**	**Conditions**
PutValency($\searrow l$)	$(\sigma \mid w_i,\beta,\theta,A) \Rightarrow (\sigma,w_i \mid \beta,\searrow l^i\theta,A)$	$\forall l' \exists! k \;\; (k,l',i) \in A$
Dist-Right($\nearrow l$)	$(\sigma \mid w_j,\beta,\theta_1 \searrow l^i \theta_2,A) \Rightarrow (\sigma \mid w_j,\beta,\theta_1\theta_2,A \cup \{(j,l,i)\})$	$\searrow l \notin \theta_1$
RShift	$(\sigma \mid w_i,\beta,\theta,A) \Rightarrow (\sigma,w_i \mid \beta,\theta,A)$	

TABLE 7.4 – Définition des transitions du système de prédiction des dépendances non-projectives droites.

La séparation des étapes de prédiction des dépendances non-projectives est essentielle pour réussir à prédire de manière plus efficace les dépendances droites. En pratique, trouver les têtes des dépendances (i.e. trouver les valences nord $\nwarrow l$ ou $\nearrow l$) est plus facile lorsque les dépendants (i.e. les valences sud $\swarrow l$ ou $\searrow l$) ont été trouvés auparavant. Les deux systèmes permettent donc de toujours prédire les valences sud avant les valences nord dans les deux cas (gauche et droite) et bénéficient donc de ces informations pour trouver les têtes des dépendances non-projectives. Tandis que la bonne prédiction des valences sud tient en majeure partie à la bonne prédiction de l'ancre correspondant à la dépendance non-projective et en particulier à l'étiquette de la dépendance assignée.

Un exemple d'application de ces deux systèmes est présenté dans le tableau 7.5. L'ensemble d'arc initial A contient la structure projective présentée par la figure 7.2 et l'ensemble A_2 de la

configuration finale contient la structure de dépendances non-projective pour la phrase « Pierre y est allé, soutenu par sa famille . » exposée dans la figure 7.1.

Transition	**Configuration**			
	([w_0],	[Il,...,.],	(),	A)
Shift	$\Rightarrow$ ([...,Il],	[y,...],	(),	A)
PutValency(↙clit-l-obj)	$\Rightarrow$ ([...,Il],	[y,...],	(↙clit-l-obj),	A)
Shift	$\Rightarrow$ ([...,y],	[est,...],	(↙clit-l-obj),	A)
Shift	$\Rightarrow$ ([...,est],	[alle,...],	(↙clit-l-obj),	A)
DistLeft(↖clit-l-obj)	$\Rightarrow$ ([...,est],	[alle,...],	(),	$A_1 = A \cup \{(4,\text{clit-l-obj},2)\}$)
Shift **(x6)**	$\Rightarrow$ ([w_0,...,.],	[],	(),	A_1)
	([w_0,... ,famille],	[.],	(),	A_1)
RShift	$\Rightarrow$ ([w_0,...,],	[famille,.],	(),	A_1)
RShift **(x3)**	$\Rightarrow$ ([...,,],	[soutenu,...],	(),	A_1)
PutValency(↘modif)	$\Rightarrow$ ([...,,],	[soutenu,...],	(↘modif),	A_1)
RShift **(x5)**	$\Rightarrow$ ([w_0],	[il,...],	(↘modif),	A_1)
DistLeft(↗modif)	$\Rightarrow$ ([w_0],	[il,...],	(),	$A_2 = A_1 \cup \{(1,\text{modif},6)\}$)

TABLE 7.5 – Séquences de transitions permettant d'obtenir la structure de dépendances non-projective de la phrase « Pierre y est allé, soutenu par sa famille. ». à partir de la structure de dépendances projective contenu dans A. La première séquence correspond à l'application du système de prédiction des dépendances non-projectives gauches et la seconde séquence correspond à l'application du système de prédiction des dépendances non-projectives droites. L'ensemble d'arcs A_1 en sortie du premier système est donné en entrée du second système.

7.3 Expérimentations

Notre méthode étant inspirée de la représentation en dépendances induite par les règles des grammaires catégorielles de dépendances, nous testons dans un premier temps notre méthode sur le CDG Treebank. Puis, dans un souci d'adaptation de notre méthode à d'autres langues et corpus en dépendances, nous proposons d'adapter des structures de dépendances standards (sans ancres) à la représentation des CDG. Nous testons donc, dans un deuxième temps, notre méthode sur des corpus dans différentes langues et n'utilisant pas la représentation des CDG.

7.3.1 CDG Treebank

Nous évaluons notre méthode sur le CDG Treebank, contenant 3 030 structures de dépendances dans lesquelles sont utilisées des dépendances projectives, des dépendances non-projectives et des ancres. Nous souhaitons d'une part comparer les résultats de notre méthode avec d'autres méthodes d'analyse par transition et d'autre part adapter des corpus standards à notre méthode.

• **Les données** sont alors exploitées sous trois formes différentes :

- Les structures d'origines du CDG Treebank sont préservées pour effectuer des expérimentations avec la méthode que nous proposons. Dans cette section nous nommons alors le corpus CDG Treebank 1.

- Les ancres des structures non-projectives sont retirées pour effectuer des expérimentations avec des méthodes d'analyse standards.

- Les ancres d'origines sont retirées et remplacées par des ancres assignées automatiquement dans le but de montrer qu'un corpus standard peut être adapté facilement à notre méthode. Le corpus résultant est nommé CDG Treebank 2.

Les ancres apparaissant dans les structures de dépendances d'origine sont spécifiques aux CDG dans le sens où elles résultent de la définition des types des grammaires catégorielles de dépendances qui ont été manuellement construits et donc linguistiquement réfléchis. Cependant la finalité des ancres des CDG est la même que celles des dépendances qui peuvent être produites à l'aide de méthode de projectivisation : proposer des dépendances projectives alternatives aux dépendances non-projectives. Nous utilisons donc la méthode de projectivisation employée par Nivre et Nilsson (2005) (initialement employée par leur méthode d'analyse pseudo-projective) sur le CDG Treebank dépourvu des ancres d'origines. L'algorithme de projectivisation est présenté dans la sous-section 2.3.2 du chapitre 2. Les dépendances projectives (en ce cas des ancres artificielles) qui en résultent ne sont alors pas substituées aux dépendances non-projectives mais ajoutées aux structures pour obtenir un corpus adoptant la représentation mixte des CDG. Nous obtenons finalement un CDG Treebank 2 très similaire au CDG Treebank 1 puisque 90,9 % des ancres sont identiques (i.e. obtiennent la même tête).

• **Le système de prédiction** que nous employons pour effectuer les étapes non-projectives de la méthode est le même que dans le chapitre précédent (section 6.3.2). Nous utilisons donc un SVM pour apprendre et prédire les transitions des systèmes. En outre, les patrons de traits sont adaptés aux systèmes mais sont dans l'ensemble également les mêmes. En ce qui concerne l'étape projective, le MaltParser (Nivre *et al.*, 2006a) est utilisé et repose également sur un système de prédiction exploitant l'efficacité des SVM.

Dans le cadre d'une comparaison avec d'autres systèmes d'analyse par transition, nous choisissons de conduire les expérimentations sur les algorithmes proposés par le MaltParser et de présenter ceux ayant les meilleurs résultats dans deux catégories différentes :

- le système *covnonproj* (Covington, 2001), proposant une analyse non-projective en un seul traitement ;

- le système *arc-eager* associé à la méthode pseudo-projective (Nivre et Nilsson, 2005) permettant de retrouver les dépendances non-projectives par déprojectivisation.

Les patrons de traits employés pour ces systèmes sont déterminés à l'aide du MaltOptimizer (Ballesteros et Nivre, 2012).

• **Les critères d'évaluation** demeurent identiques à ceux exposés dans le chapitre précédent. Nous procédons à des évaluations croisées et chaque partie de test est pré-étiquetée grammaticalement par Melt (voir la section 6.5) avant analyse. La comparaison des scores d'analyse en dépendances se fait alors suivant les critères classiques : LA, LAS et UAS. Dans un souci d'équité, les scores d'attachement comprennent uniquement les résultats sur les dépendances projectives et sur les dépendances non-projectives mais n'incluent pas les scores sur les ancres dans le cas de notre méthode. En outre, les ponctuations ne sont pas incluses dans le calcul des différents scores.

• **Les résultats** des expérimentations sont présentés dans le tableau 7.6. Notons, dans un premier temps que les scores obtenus par la méthode que nous proposons (3-4), sur les dépendances projectives, surpassent (+0,9/0,8 LAS) ceux de la méthode *covnonproj* (1) mais équivalent (-0,1/0,2 LAS) les résultats de la méthode pseudo-projective (2). La méthode *covnonproj* (1), joignant la prédiction des dépendances projectives et des dépendances non-projectives dans un même système, atteint des scores inférieurs sur les dépendances projectives alors que la méthode projective *arc-eager* permet d'obtenir de bons scores sur les dépendances projectives dans les deux situations dans lesquelles elle est employée, (2) et (3-4). En (2) il s'agit de prédire les dépendances projectives d'origine et les dépendances projectives contenant une information, encodée dans leurs étiquettes, sur les dépendances non-projectives auxquelles elles sont associées. Tandis qu'en (3-4) il s'agit de prédire les dépendances projectives et les ancres sans différenciation des unes par rapport aux autres. Dans le premier cas (2), on obtient alors des scores très légèrement supérieurs (mais pas significativement) pour les dépendances projectives mais la précision des étiquettes (LA) est inférieure (-2,0) pour les dépendances non-projectives et occasionnent des scores très inférieurs en LAS (≈-13,4) et UAS (≈-12,9) pour ces dépendances. Dans le cas de la méthode que nous proposons, la bonne prédiction des ancres et en particulier de leurs étiquettes syntaxiques[4] semble donc profitable à la reconnaissance des dépendances non-projectives. En résumé, notre méthode permet donc d'obtenir de meilleurs scores sur les dépendances non-projectives et également sur l'ensemble des dépendances.

	Toutes dép.			**Dép. Projectives**			**Dép. Non-Proj.**		
	LA	**UAS**	**LAS**	**LA**	**UAS**	**LAS**	**LA**	**UAS**	**LAS**
(1) **covnonproj**	82,2	85,5	78,0	82,8	86,2	78,7	68,7	68,7	62,7
(2) **pseudo-proj** [5]	83,6	85,9	78,7	**84,1**	**87,0**	**79,7**	73,5	56,9	53,5
(3) **non-projLR** (CDGTbk1)	**83,7**†	**86,3**‡	**79,1**‡	**84,1**†	86,9†	79,6†	**75,5**†	70,2$^{\downarrow}$	66,3‡
(4) **non-projLR** (CDGTbk2)	**83,7**†	86,2†	79,0‡	**84,1**†	86,9†	79,5†	**75,5**†	**70,5**$^{\downarrow}$	**66,7**‡

TABLE 7.6 – Résultats des expérimentations sur le CDG Treebank. Notre méthode (3-4), ici nommée non-projLR, est comparée aux méthodes *covnonproj* et pseudo-projective (+*arc-eager*) du MaltParser. Le symbole † signifie que les scores de notre méthode (3-4) sont significativement supérieurs (à 0,5 %, conformément au test de Student) à la méthode *covnonproj* et le symbole ↓ signifie que les scores sont significativement supérieurs à la méthode pseudo-projective (+*arc-eager*). Le symbole ‡ signifie que les scores sont significativement supérieurs aux deux méthodes.

En examinant plus en détail les résultats sur les dépendances non-projectives, il est également à noter que des différences persistent entre les dépendances droites et les dépendances gauches (comme observé dans le chapitre précédent, section 6.5). En effet, les dépendances non-projectives gauches atteignent des scores supérieurs (75,0 % LAS) par rapport aux dépendances non-projectives droites (42,7 % LAS). Néanmoins ces scores sont supérieurs à ceux obtenus par la méthode que nous avons proposée dans le chapitre précédent et en particulier sur les dépendances non-projectives droites (+24,6). Nous savons donc qu'il est nécessaire, dans le cadre du formalisme

[4]Les systèmes non-projectifs droits et gauches exploitent en particulier, dans leurs traits, des informations sur les étiquettes des ancres prédites en amont pour prédire les dépendances non-projectives. Les résultats sur les ancres, après l'étape projective, sont de 75,5 % en LA, 89,0 % en UAS et 72,1 % en LAS dans le cadre de notre méthode sur le CDGTbk1 (3) et de 75,5 % en LA, 87,5 % en UAS et 72,5 % en LAS sur le CDGTbk2 (4). Le score LA des dépendances ancres et des dépendances non-projectives est le même puisqu'il s'agit du pourcentage de mots pour lesquels la bonne étiquette a été trouvée (quelle que soit la sorte de la dépendance : projective ou non-projective). Ce score bénéficie donc tout d'abord de la bonne prédiction des ancres.

[5]La méthode pseudo-projective est appliquée avec l'option « path » pour la projectivisation et la déprojectivisation

sur lequel nous travaillons, d'effectuer une analyse de droite à gauche pour trouver plus efficacement les dépendances non-projectives droites. Il est en effet très important de prédire le dépendant de la dépendance avant sa tête et ce dépendant peut être lui-même prédit plus efficacement si une ancre avec la bonne étiquette fut prédite lors de l'étape projective. Les scores éloignés des dépendances non-projectives gauches et droites découlent assurément des scores obtenus sur les ancres. Seulement 51,4 % des dépendances non-projectives droites reçoivent la bonne étiquette de dépendance (LA) tandis que ce score atteint 84,2 % pour les dépendances non-projectives gauches. Nous supposons que la sous-représentation des dépendances non-projectives droites dans les données[6] est une des explications principales de leurs faibles scores. Cependant, l'analyse des résultats en fonction des étiquettes de dépendances nous montre que même les dépendances associées à des dépendances fréquentes obtiennent des scores faibles. De plus, sur l'ensemble des dépendances (projectives et non-projectives) nous remarquons que les scores sur les dépendances droites sont également inférieurs aux scores sur les dépendances gauches. Le problème soulève alors de nouvelles interrogations quant à l'utilisation de systèmes d'analyse par transition effectuant communément les analyses de gauche à droite.

Par ailleurs, nous avons expérimenté notre méthode sur les CDG Treebank 1 et 2 dans le but de comparer leurs résultats. Les deux corpus étant très similaires, les résultats sont équivalents, bien que très légèrement supérieurs sur les dépendances non-projectives dans le cas du CDG Treebank 2. L'utilisation des ancres artificielles dans le corpus est donc semblable à l'utilisation des ancres d'origines. En conséquence, la méthode est adéquate et peut être adaptée pour l'analyse d'autres langues à partir de corpus en dépendances standards.

Notons que les expérimentations étant effectuées ici sur les données du CDG Treebank, les scores obtenus sont difficilement comparables à ceux obtenus pour le français sur les données du French Treebank (FTB) (Abeillé *et al.*, 2003) en utilisant le MaltParser (scores supérieurs à 87 % en LAS). Le CDG Treebank, d'une part contient beaucoup moins de phrases que le FTB et d'autre part suit un schéma d'annotation dont le jeu d'étiquettes syntaxiques est beaucoup plus étendu[7] et comprenant un nombre important de fonctions syntaxiques pouvant être associées à des dépendances non-projectives.

7.3.2 Universal Dependency Treebank

Dans le but d'évaluer les performances de notre méthode pour diverses langues, nous appliquons la technique présentée précédemment pour le CDG Treebank 2 à d'autres corpus en dépendances ne comprenant pas la gestion des ancres.

• **Les données** que nous exploitons pour ces expérimentations proviennent du *Universal Dependency Treebank* (UDT) (McDonald *et al.*, 2013). Le UDT rassemble des corpus en dépendances pour différentes langues dont les schémas d'annotation (grammatical et syntaxique) suivent les mêmes lignes (voir la section 2.4 du chapitre 2). Entre autre, le schéma d'annotation ne fut pas limité par une contrainte de projectivité. En conséquence, certains corpus comprennent des dépendances non-projectives. Nous décidons d'effectuer les expérimentations sur les corpus dont le pourcentage de phrases non-projectives est le plus élevé (bien que les taux soit parfois relativement bas). Il s'agit des corpus pour : l'allemand, l'espagnol, le français, l'italien et le suédois. Les statistiques des corpus sont regroupées dans le tableau 7.7.

[6] Les dépendances non-projectives droites représentent 25,6 % des dépendances non-projectives.
[7] 117 étiquettes syntaxiques sont utilisées dans le CDG Treebank contre 28 pour le French Treebank.

		DE	ES	FR	IT	SV
Corpus d'entrainement	**Unités lexicales**	229 808	332 979	308 155	132 045	59 347
	% non-proj.	0,28	0,27	0,86	0,37	0,37
	Phrases	14 118	14 138	14 511	6 389	4 447
	% non-proj.	9,15	5,09	12,27	4,87	4,03
Corpus de test	**Unités lexicales**	13 964	7 392	6 052	8 125	18 278
	% non-proj.	2,48	0,58	0,73	0,44	0,28
	Phrases	1000	300	300	400	1 219
	% non-proj.	20,6	11,0	8,33	6,25	3,45

TABLE 7.7 – Statitiques des corpus allemand (DE), espagnol (ES), français (FR), italien (IT) et suédois (SV) du *Universal Dependency Treebank*. Le calcul des unités lexicales ne comprend pas les signes de ponctuation.

• **Les critères d'évaluation** des expérimentations comprennent le LA, le LAS et le UAS. Chaque score calculé ne tient pas compte des signes de ponctuations. Les expérimentations comprennent uniquement l'analyse en dépendances (les étiquettes grammaticales correctes sont conservées pour les phases d'entraînement et les phases de test).

Par ailleurs, nous présentons également les résultats obtenus avec différentes méthodes proposées par le MaltParser. Dans le cas des analyses non-projectives (NP) nous employons les méthodes :

- *stacklazy* ;
- *covnonproj* ;
- et *stackeager*.

Dans le cas des analyses pseudo-projectives (PP), nous employons les méthodes :

- *stackproj* ;
- *nivreeager* (nom de l'option d'analyse du MaltParser correspondant à la méthode *arc-eager*) ;
- et *nivrestandard* (nom de l'option d'analyse du MaltParser correspondant à la méthode *arc-standard*).

Notre méthode pouvant exploiter le potentiel de n'importe quelle système d'analyse projectif comme première étape, nous choisissons de présenter les résultats obtenus en utilisant les mêmes méthodes que pour les analyses pseudo-projectives.

• **Les résultats** sont présentés dans les tableaux 7.8, 7.9, 7.10, 7.11 et 7.12. Les premiers résultats (tableau 7.8) proviennent des expérimentations sur le corpus allemand du UDT qui est également le corpus contenant le plus de dépendances non-projectives dans son ensemble de test. Les résultats sur les dépendances non-projectives ont donc un impact important pour les résultats sur l'ensemble des dépendances. On constate que les meilleurs scores sont obtenus par notre méthode lorsqu'elle est combinée à l'algorithme projectif *nivrestandard*. Les scores sont relativement proches de ceux obtenus par l'analyse pseudo-projective associée à la méthode *nivrestandard* mais tout de même supérieurs pour les dépendances projectives (+0,3 LAS) et non-projectives (+0,6 LAS) pour obtenir globalement des scores intéressants.

		Toutes dép.			Dép. Projectives			Dép. Non-Proj.		
		LA	**UAS**	**LAS**	**LA**	**UAS**	**LAS**	**LA**	**UAS**	**LAS**
NP	**stacklazy**	86,6	83,9	78,1	87,1	84,8	79,0	**65,0**	48,8	40,5
	covnonproj	85,1	83,5	77,0	85,8	84,7	78,2	56,9	35,8	30,3
	stackeager	86,0	82,9	77,1	86,6	83,8	78,1	62,7	46,0	37,3
PP	**stackproj**	86,7	84,5	78,5	87,3	85,4	79,5	63,0	48,3	39,0
	nivreeager	85,9	84,2	77,4	86,5	85,2	78,5	61,0	44,8	36,1
	nivrestandard	86,8	**84,7**	78,5	87,4	**85,6**	79,5	62,7	50,3	40,2
LR	**stackproj**	86,4	84,0	78,0	87,0	84,9	79,0	63,0	46,2	37,3
	nivreeager	86,1	83,9	77,5	86,7	84,9	78,5	62,1	47,7	39,0
	nivrestandard	**87,0**	**84,7**	**78,9**	**87,6**	**85,6**	**79,8**	63,3	**50,6**	**40,8**

TABLE 7.8 – Résultats des expérimentations sur le corpus allemand du UDT Treebank.

		Toutes dép.			Dép. Projectives			Dép. Non-Proj.		
		LA	**UAS**	**LAS**	**LA**	**UAS**	**LAS**	**LA**	**UAS**	**LAS**
NP	**stacklazy**	**87.7**	**85.0**	80.6	**87.9**	**85.4**	**81.0**	58.1	11.6	9.3
	covnonproj	86,4	83,7	79,6	86,6	84,1	80,0	55,8	7,00	7,0
	stackeager	87,5	84,9	80,5	87,7	85,3	80,9	65,1	14,0	14,0
PP	**stackproj**	87,4	84,7	80,2	87,5	85,2	80,6	62,8	9,3	7,0
	nivreeager	86,8	84,3	80,0	87,0	84,8	80,4	65,1	4,7	4,7
	nivrestandard	87,6	84,9	80,4	87,8	**85,4**	80,9	62,8	7,0	4,7
LR	**stackproj**	87,3	84,7	80,2	87,4	85,0	80,5	**72,1**	**30,2**	**27,9**
	nivreeager	87,5	84,6	**80,7**	87,7	85,0	**81,0**	62,8	18,6	18,6
	nivrestandard	87,3	84,6	80,1	87,4	84,9	80,4	**72,1**	**30,2**	**27,9**

TABLE 7.9 – Résultats des expérimentations sur le corpus espagnol du UDT Treebank.

Les résultats sur les corpus espagnol (tableau 7.9) et français (tableau 7.10) sont relativement comparables les uns avec les autres. En effet, les conditions d'expérimentation sont assez similaires dans le sens où les corpus d'entraînement comportent un nombre important de dépendances non-projectives mais les corpus de test comptent très peu de dépendances non-projectives. On remarque alors que les résultats sur les dépendances non-projectives atteignent les meilleurs scores dans le cadre de notre méthode (au minimum +13,9 pour l'espagnol et +4,5 pour le français en LAS), lorsque celle-ci est combinée avec les méthodes d'analyse projective *stackproj* et *nivrestandard*. Cependant les résultats sur les dépendances projectives sont très variables. Dans le cas de l'espagnol, les meilleurs scores sur les dépendances projectives sont obtenus par la méthode non-projective *stacklazy*. Cependant la méthode *nivreeager* dans le cadre de notre système obtient les mêmes scores en LAS. Alors bien que cette méthode obtienne des scores plus bas que les deux autres (*stackproj* et *nivrestandard*) sur les dépendances non-projectives, son score global en LAS est le plus élevé (+0,1 par rapport au second meilleur en LAS). Néanmoins, le fait que les scores soient quasiment équivalents et que les expérimentations soient effectuées sur seulement un ensemble de test fait que les résultats ne peuvent pas être considérés comme significativement supérieurs. Dans le cas du français, les résultats sont également très similaires pour les dépendances projectives. La méthode *nivreeager* obtient les meilleurs résultats, étant quasiment équivalents entre le système

		Toutes dép.			Dép. Projectives			Dép. Non-Proj.		
		LA	**UAS**	**LAS**	**LA**	**UAS**	**LAS**	**LA**	**UAS**	**LAS**
NP	**stacklazy**	85,6	84,3	78,9	85,8	84,7	79,3	63,6	27,3	20,5
	covnonproj	85	83,5	78,4	85,2	84,1	78,9	52,3	9,1	9,1
	stackeager	84,7	83,5	77,9	84,9	83,9	78,3	59,1	22,7	20,5
PP	**stackproj**	85,4	84,1	78,4	85,5	84,5	78,9	**70,5**	25,0	18,2
	nivreeager	86,2	**84,6**	**79,5**	86,3	**85,1**	**79,9**	65,9	15,9	13,6
	nivrestandard	85,5	84	78,5	85,6	84,5	78,9	**70,5**	25,0	18,2
LR	**stackproj**	85,6	83,9	78,5	85,7	84,3	78,9	**70,5**	**29,5**	**25,0**
	nivreeager	**86,3**	84,4	79,4	**86,5**	84,8	79,8	65,9	18,2	15,9
	nivrestandard	85,4	83,7	78,4	85,5	84,1	78,8	**70,5**	**29,5**	**25,0**

TABLE 7.10 – Résultats des expérimentations sur le corpus français du UDT Treebank.

pseudo-projectif et notre système. Les résultats globaux sont donc également équivalents car le très faible nombre de dépendances non-projectives dans les données de test ne permettent pas de faire une différence dans l'ensemble bien que les scores sur les dépendances non-projectives soient supérieurs avec cette méthode (+2,3 LAS).

Pour finir, les résultats sur les corpus italien (tableau 7.11) et suédois (tableau 7.12) sont également très variables. Ces deux corpus comportent un nombre bien plus faible de dépendances non-projectives dans les données d'entraînement que les corpus précédents mais également un très faible nombre de dépendances non-projectives dans les données de test. On constate que les résultats obtenus avec notre méthode sur les dépendances non-projectives sont inférieurs aux autres méthodes. En particulier, on remarque que les scores en LA (bonne prédiction des étiquettes) sont plus faibles dans le cadre de notre méthode (-5,5 pour l'italien et -1,9 pour le suédois par rapport aux meilleurs scores), ce qui signifie que les ancres sont moins facilement trouvées. Le fait de considérer les ancres comme des dépendances projectives semble donc être un désavantage dans ce cadre là, pour la prédiction de ces ancres et par conséquence pour la prédiction des dépendances non-projectives. Le faible nombre de dépendances non-projectives (et par équivalence, le faible nombre d'ancres) fait que les ancres qui leurs correspondent sont probablement noyées dans l'ensemble des dépendances projectives de même rôle syntaxique durant la phase d'entraînement et donc plus difficilement différentiable par rapport à celles-ci. Si les ancres étaient prédites avec les même étiquettes que dans le cas des méthodes pseudo-projective[8] alors elles obtiendraient les même scores en LA, ce qui permettrait d'améliorer les scores en UAS et LAS. Il serait donc intéressant dans ce cas là d'intégrer une option permettant de choisir de prédire les ancres en tant qu'ancres, i.e. avec des étiquettes spécifiques indiquant leur statut d'ancre, ou en tant que dépendances projectives, i.e. avec les mêmes étiquettes. Néanmoins, l'observation des scores sur les dépendances projectives nous montre que le choix d'étiquetage des ancres a un impact sur la prédiction des dépendances projectives. Par exemple, ici, on observe que la méthode pseudo-projective obtient de meilleurs résultats que notre méthode (dans le cas d'une combinaison avec l'algorithme *nivreeager* pour les deux) sur les dépendances projectives du corpus italien (-0,2 LAS) mais on observe également l'inverse sur les dépendances projectives du corpus suédois (+0,1 LAS).

[8]Pour une même dépendance projective correspondant à une ancre et dont le rôle syntaxique est noté l, celle-ci sera étiquetée l dans le cas de notre méthode (elle est donc confondue avec les dépendances projectives de même rôle syntaxique) mais sera étiquetée λl dans le cadre de la méthode pseudo-projective (elle y est donc différenciée des dépendances projectives de même rôle) où λ est l'information encodée par la méthode selon l'option choisie (le MaltParser propose les options "*head*" et "*path*" pour retrouver les dépendances non-projectives).

En conséquence, notre méthode obtient des résultats globaux légèrement supérieurs sur le corpus suédois (+0,1 LAS) bien que les résultats sur les dépendances non-projectives soient plus faibles (-5,7 LAS).

		Toutes dép.			Dép. Projectives			Dép. Non-Proj.		
		LA	**UAS**	**LAS**	**LA**	**UAS**	**LAS**	**LA**	**UAS**	**LAS**
NP	**stacklazy**	89,9	86,0	82,6	90,0	86,2	82,8	66,7	**38,9**	**33,3**
	covnonproj	89,5	85,7	82,7	89,6	86,0	83,0	55,6	25,0	25,0
	stackeager	89,5	85,4	82,0	89,6	85,6	82,2	**69,4**	33,3	30,6
PP	**stackproj**	90,2	86,2	82,9	90,3	86,4	83,1	66,7	36,1	30,6
	nivreeager	90,5	**86,4**	**83,3**	90,6	**86,7**	**83,6**	58,3	22,2	22,2
	nivrestandard	90,1	86,1	82,8	90,2	86,3	83,1	66,7	36,1	30,6
LR	**stackproj**	90,0	85,8	82,5	90,1	86,1	82,8	63,9	33,3	30,6
	nivreeager	**90,6**	86,2	83,1	**90,7**	86,5	83,4	61,1	30,6	27,8
	nivrestandard	90,1	85,9	82,6	90,2	86,1	82,8	63,9	33,3	30,6

TABLE 7.11 – Résultats des expérimentations sur le corpus italien du UDT Treebank.

		Toutes dép.			Dép. Projectives			Dép. Non-Proj.		
		LA	**UAS**	**LAS**	**LA**	**UAS**	**LAS**	**LA**	**UAS**	**LAS**
NP	**stacklazy**	88,2	85,6	81,4	88,4	85,8	81,6	30,8	13,5	9,6
	covnonproj	88,1	85,2	81,5	88,3	85,4	81,7	28,8	5,8	3,8
	stackeager	88,8	86,4	82,3	89,0	86,6	82,5	30,8	7,7	1,9
PP	**stackproj**	**89,5**	86,7	83,0	**89,6**	86,9	83,2	**40,4**	17,3	13,5
	nivreeager	88,6	86,0	82,1	88,7	86,2	82,3	38,5	13,5	11,5
	nivrestandard	89,1	86,5	82,9	89,2	86,7	83,1	**40,4**	**23,1**	**19,2**
LR	**stackproj**	88,9	**86,8**	82,6	89,0	**87,0**	82,8	38,5	17,3	13,5
	nivreeager	89,3	**86,8**	**83,1**	89,5	**87,0**	**83,3**	36,5	11,5	9,6
	nivrestandard	88,6	85,9	82,1	88,8	86,1	82,3	34,6	13,5	9,6

TABLE 7.12 – Résultats des expérimentations sur le corpus suédois du UDT Treebank.

7.4 Conclusion

Nous avons présenté dans ce chapitre un processus d'analyse par transition en trois étapes permettant dans un premier temps de répondre aux problèmes soulevés dans le chapitre précédent. Parmi ces trois étapes, nous proposons en particulier deux systèmes d'analyse par transitions que nous avons mis en œuvre et permettant de prédire les dépendances non-projectives gauches et droites à partir des structures de dépendances projectives produites au préalable par n'importe quel système projectif. La séparation des étapes de prédiction des dépendances projectives et non-projectives permet d'obtenir des scores d'analyse en dépendances bien supérieurs à ceux obtenus précédemment. Dans le cadre d'expérimentations sur le CDG Treebank, on remarque que, d'une

part les scores sur les dépendances non-projectives sont meilleurs et que d'autre part, les dépendances non-projectives droites sont mieux prédites grâce à l'étape de prédiction spécifique à ces dépendances parcourant la phrase de droite à gauche. La méthode que nous proposons permet également d'obtenir des scores plus hauts que ceux obtenus par les méthodes du MaltParser.

Par ailleurs, il était nécessaire de pouvoir appliquer cette méthode à des corpus n'étant pas annotés selon le même schéma que le CDG Treebank, i.e. ne comprenant pas d'ancres. Nous avons donc montré qu'il est possible d'obtenir des corpus conformes à cette représentation à partir de corpus standards en ajoutant automatiquement les ancres à l'aide d'une méthode de projectivisation des dépendances non-projectives (ici le module de projectivisation de la méthode pseudo-projective de Nivre et Nilsson (2005)). Nous avons donc expérimenté notre méthode sur les corpus du *Universal Dependency Treebank* (McDonald *et al.*, 2013) et comparé les résultats avec ceux obtenus par les méthodes pseudo-projectives et non-projectives du MaltParser. Les résultats sont relativement variables selon les ensembles de données. Cependant, notre méthode permet d'améliorer les scores d'analyse en dépendances lorsque les données d'entraînement sur les dépendances non-projectives sont suffisantes. En outre, les scores sur les dépendances projectives semblent dépendre de la façon dont sont prédites les ancres, de manière indifférenciée avec les dépendances projectives ou non. Selon les corpus, il serait donc intéressant de faire varier le mode d'étiquetage des ancres pour optimiser la prédiction des dépendances projectives.

Globalement, la méthode que nous proposons permet d'améliorer les scores d'analyse en dépendances et particulièrement les scores sur les dépendances non-projectives. Elle est, de plus, adaptable à tout corpus en dépendances non-projectif.

8

Conclusion

8.1 Contributions

Cette thèse rassemble des travaux distincts mais répondants à des problèmes reliés dans le domaine de l'analyse en dépendances non-projective. D'une part, on constate qu'il existe aujourd'hui peu de corpus en dépendances librement disponibles pour le français construits (non-convertis) et contenant une part significative de dépendances non-projectives. Bien que le français comprenne un certain nombre de phénomènes discontinus, ils ne sont pas toujours pris en compte ou apparaissent très peu dans les ensembles de phrases choisis pour l'annotation. Dans le domaine de la syntaxe, les grammaires catégorielles de dépendances permettent de gérer (Dekhtyar et Dikovsky, 2004, 2008) les dépendances non-projectives dans les langues. En outre, le développement d'une grammaire du français (Dikovsky, 2011) associé à l'utilisation d'un analyseur basé sur les grammaires catégorielles de dépendances (inclus dans l'environnement intégré CDG Lab (Béchet *et al.*, 2014)) ont engagé des travaux visant la construction de corpus en dépendances non-projectifs pour le français. Les premiers corpus en dépendances pour le français formés à l'aide de la grammaire catégorielle du français proposent la description d'un large échantillon de phénomènes discontinus parmi l'ensemble des dépendances employées pour l'annotation de ces corpus. Néanmoins, dans le cadre du développement de large corpus en dépendances non-projectifs pour le français, le processus d'annotation du CDG Lab reste contraignant pour les annotateurs. Il demande un travail manuel relativement long et fastidieux aux annotateurs. En outre, la totalité des corpus en dépendances (CDG Treebank) déjà développés représente un ensemble de phrases annotées suffisant pour effectuer des expérimentations sur ces données et obtenir des résultats intéressants. Il est donc possible à partir de ces données (cohérentes avec le schéma d'annotation induit par les grammaires catégorielles de dépendances) d'intégrer des méthodes statistiques dans le processus d'analyse du CDG Lab permettant de rendre plus efficace le travail des annotateurs en aval.

D'autre part, dans le domaine de l'analyse en dépendances, il existe aujourd'hui un nombre conséquent de méthodes dirigées par les données rapides et efficaces. La disponibilité de large corpus en dépendances pour diverses langues a permis l'émergence de ces méthodes. Dans le cadre des méthodes dirigées par les données, les systèmes par transitions, en particulier, sont rapides pour l'entraînement comme pour l'analyse en dépendances. En outre, ces systèmes permettent également de gérer les dépendances non-projectives. Dans la réalisation de certaines tâches du

traitement automatique des langues naturelles, telles que pour les systèmes de question-réponse, il est important de pouvoir utiliser des systèmes d'analyse en dépendances qui soient efficaces en terme de rapidité et de précision sur les dépendances, et en particulier sur les dépendances non-projectives. Cependant, les dépendances non-projectives sont souvent des dépendances de longue distance difficiles à prédire pour les systèmes par transition. En effet, les scores des dépendances non-projectives sont toujours inférieurs aux scores des dépendances projectives sur un même ensemble de données. Il est donc nécessaire de travailler à l'amélioration de la prédiction des dépendances non-projectives.

La partie II de cette thèse rassemble les travaux que nous avons effectués dans le cadre de la mise en place d'un processus d'annotation en dépendances non-projectif pour le français. Le processus d'annotation du CDG Lab étant contraignant pour les annotateurs, nous avons proposé d'intégrer un processus automatique de pré-annotation allégeant le travail des annotateurs. En effet, le processus d'annotation de base offrait deux choix aux annotateurs : employer un processus autonome d'analyse en dépendances basé sur la grammaire catégorielle du français qui aboutit fréquemment à un échec de l'analyse dû à l'espace de recherche trop important, ou remplir manuellement un tableau (appelé tableau de sélection des têtes) permettant la pré-annotation des phrases avant l'analyse. L'emploi du tableau de sélection est efficace pour réduire l'espace de recherche de l'analyseur mais peut être long à remplir pour les annotateurs. Dans le cadre de nos travaux, nous souhaitions remplacer ce tableau manuel par une sélection automatique. Nous avons montré dans un premier temps qu'une sélection des têtes automatique (étiquetage syntaxique) pouvait être efficace et qu'elle était utile à l'analyse en dépendances. En effet, elle permet de réduire l'espace de recherche de l'analyseur sans subir une perte trop importante de précision. Nous avons alors cherché à mettre en place un processus intégralement autonome de pré-annotation en dépendances. Nous avons tout d'abord proposé une méthode de prédiction jointe des classes grammaticales et des étiquettes syntaxiques prenant en compte toutes les possibilités de segmentation des phrases à analyser. Cependant, la segmentation incertaine des phrases n'a pas permis une sélection efficace des étiquettes syntaxiques et ainsi une réduction suffisante de l'espace de recherche pour atteindre des scores d'analyse en dépendances intéressants. Nous avons alors proposé un processus de pré-annotation automatique en plusieurs étapes. Le processus comprend la segmentation automatique des phrases en mots, l'étiquetage grammaticale de ces mots et leur étiquetage syntaxique. À partir des informations apportées par ce processus, l'analyseur en dépendances recherche les analyses possibles des phrases données à analyser dans un espace de recherche réduit mais plus précis que précédemment. Nous obtenons alors un processus complet de pré-annotation en dépendances suffisamment efficace en terme de rapidité et de précision pour profiter aux annotateurs. L'annotation de corpus en dépendances non-projectifs est plus agréable et efficace pour les annotateurs.

La partie III de cette thèse rassemble nos derniers travaux, effectués dans le cadre de l'étude d'un système d'analyse par transition. Nous proposons, dans un premier chapitre, d'évaluer l'efficacité d'un système par transition adapté à la représentation mixte projective/non-projective induite par les grammaires catégorielles de dépendances. Ce système permet de gérer les dépendances projectives, non-projectives et les ancres employées dans cette représentation. Nous avons étudié ce système, mis en évidence certaines limitations et proposé des alternatives permettant d'améliorer le système. Les résultats obtenus sur les données du CDG Treebank montrent la pertinence des évolutions que nous avons proposées pour ce système mais soulèvent également d'autres interrogations. Les scores sont intéressants mais restent en dessous des scores que l'on peut obtenir avec un analyseur par transition standard tel que le MaltParser (Nivre *et al.*, 2006a). La prédiction simultanée des dépendances projectives et non-projectives n'est pas suffisamment efficace dans le cadre de cette méthode. En outre, nous remarquons qu'en ce qui concerne la prédiction des dépendances non-projectives, nous obtenons des écarts très importants entre les scores sur les dépendances droites et les scores sur les dépendances gauches. Dans un second chapitre, nous proposons

alors un système en trois étapes permettant de pallier à ces problèmes. Nous proposons d'effectuer une analyse en dépendances par transition en séparant les étapes de prédiction des dépendances projectives, des dépendances non-projectives gauches et des dépendances non-projectives droites. Ce processus permet d'obtenir de bien meilleurs scores sur les mêmes données que dans le chapitre précédent et également plus élevés que ceux des méthodes du MaltParser. Nous obtenons des résultats équivalents sur les dépendances projectives. De plus, par la prédiction séparée des dépendances non-projectives gauches et droites, nous obtenons des scores supérieurs aux méthodes standards et nous réduisons l'écart de précision entre les dépendances gauches et droites. Bien que ce système soit adapté à la représentation en dépendances mixte des CDG, nous montrons également qu'il peut être employé pour l'analyse de corpus en dépendances standards. Grâce à l'outil de projectivisation (Nivre et Nilsson, 2005) du MaltParser nous ajoutons des ancres artificielles aux structures de dépendances des corpus UDT (McDonald *et al.*, 2013) et effectuons des analyses avec notre système en trois étapes. Les résultats des analyses montrent que le système est efficace dans le cas où les corpus comprennent un nombre suffisant de dépendances non-projectives. Nous avons donc proposé un système efficace en trois étapes linéaires pour l'analyse en dépendances non-projective.

À travers nos travaux, nous avons d'une part contribué à alléger le travail des annotateurs dans le cadre du développement de corpus en dépendances non-projectifs et d'autre part amélioré la prédiction des dépendances non-projectives pour les systèmes d'analyses par transition.

8.2 Perspectives

Dans le cadre de la mise en place du processus de pré-annotation automatique du CDG Lab nous avons pu observer que la sélection automatique des étiquettes syntaxiques permet d'atteindre des scores d'analyse en dépendances intéressants. Cependant, ces scores sont calculés à partir de la meilleure structure de dépendances obtenue parmi les différentes structures en sortie de l'analyseur. Et cette meilleure structure est trouvée sachant la structure correcte d'origine. Dans le cadre du processus automatique d'analyse en dépendances du CDG Lab, nous souhaiterions mettre en place un système de tri des structures de dépendances de la même manière que dans les travaux effectués par Clark et Curran (2007) sur les CCG. L'analyse de toutes les structures possibles pour une phrase étant relativement longue il serait nécessaire d'effectuer un tri à partir des forêts de dépendances rassemblées par l'analyseur à l'aide de la matrice triangulaire des types (voir la section 4.5 du chapitre 4). En combinant notre méthode de pré-annotation syntaxique en amont de l'analyse et une méthode de tri des structures de dépendances en aval de l'analyse, nous espérons obtenir des scores d'analyse en dépendances avec les CDG atteignant ceux des méthodes d'analyse standards et allégeant toujours le travail des annotateurs.

D'autre part, les scores obtenus sur le CDG Treebank sont intéressants mais n'atteignent pas les scores qu'il est possible d'obtenir sur un corpus plus large tel que le French Treebank. Dans le but d'améliorer les prédictions des méthodes statistiques existantes et futures du CDG Lab, nous souhaiterions investiguer des méthodes de clustering telles que celles utilisées par Seddah *et al.* (2013a) qui permettent d'augmenter les scores sur des corpus de petite taille. Nous pourrions également envisager la conversion de corpus standards vers la représentation en dépendances induites par les grammaires catégorielles de dépendances dans le but de disposer de large corpus en dépendances suivant cette représentation.

Enfin, dans le cadre de l'élaboration du système par transition améliorant la prédiction des dépendances non-projectives, nous avons remarqué que les scores obtenus par notre méthode sont plus élevés que ceux des méthodes standards quand le nombre de dépendances dans les corpus

d'entraînements et de tests est suffisamment grand. Nous souhaitons évaluer cette méthode à travers des expérimentations sur des corpus en dépendances contenant un nombre plus important de dépendances non-projectives et dans le cadre d'évaluations croisées. Nous souhaitons également approfondir l'étude des dépendances non-projectives par des évaluations comparant les résultats obtenus sur les dépendances gauches et sur les dépendances droites. Dans ce cadre, nous évaluerons l'utilisation inversée des méthodes standards d'analyse en dépendances sur différents corpus en dépendances.

A

Types de dépendances des CDG

Ce que l'on appelle les types de dépendances des grammaires catégorielles de dépendances (CDG) sont des types primitifs tels que présentés dans la section 4.3 du chapitre 4. Ils correspondent aux noms des dépendances utilisées dans les représentations en dépendances induites par les CDG. Ils correspondent également à ce que nous appelons les « étiquettes syntaxiques » des mots dans nos travaux sur la pré-annotation automatique dans les CDG (chapitre 5). Dans le cas de la grammaire de dépendances du français, 117 types de dépendances différents sont exploités. Ces types peuvent également être classés dans 39 groupes de dépendances différents. Le tableau ci-dessous expose la correspondance entre les types de dépendances et leurs groupes.

Groupes de dépendances	Types de dépendances				
AGENT	Agent				
AGGR	a_aggr	aggr	c_aggr	n_aggr	p_aggr
APPOS	appos				
APPROX	approx				
ATTR	attr				
AUX	aux aux-d aux-o	aux-a aux-d-g	aux-A aux-g	aux-a-A aux-g-A	aux-a-d aux-l
CIRC	circ				
CLAUS	claus				
CLIT	0-clit-l 0-clit-g	clit-3d-obj clit-l-obj	clit-a-obj clit-g-obj	clit-copul	clit-d-obj
COMPAR	compar				
CONJ	conj-tel	aggr-conj	conj-dont	conj-que	conj-comme
COORD	coordi	coordpz	coordv		
COPRED	copred				
COPUL	a_copul c_copul	n_copul copul	a_copul-d a_copul-A	a_copul-g n_copul-g	a_copul-g-A
CORREL	correl				
DEICT	deict				
DET	det	det-p			
EMPHAT	explet	emphat	qu-emphat	junc-emphat	
GER	dep-ger				
INF	inf inf-d	inf-a inf-g	inf-A inf-o	inf-a-A pre-inf	inf-a-d pre-inf-d
INTERROG	interrog				
JUNC	cit	coord	junc	junc-spec	
MODIF	modif				
NEG	neg	restr-neg	neg-emphat	compos-neg	
OBJ	a-obj l-obj	a-obj-d o-obj	a-obj-g qa-obj	d-obj	g-obj
PRED	pred				
PREFIX	pre-attr				
PREPOS	prepos-A prepos-o	prepos-d	prepos-g	prepos-l	prepos-sel
PUNCT	cl lpar xl	cm qu	ds rp	fs rpar	lp sc
QUANT	quant				
QUANTIF	quantif				
REFLEX	auto-ref	reflex			
REL	dist-rel	loc-rel			
RESTRICT	restrict				
SELECT	select				
SENT	S				
VOCATIVE	vocative				

B

Les classes grammaticales de la CDGFr

Dans les grammaires catégorielles de dépendances, les types[1] sont associés à des classes grammaticales qui sont elles-mêmes associées aux mots du lexique de chaque langue. L'ensemble des classes grammaticales est analogue à un jeu d'étiquettes morphosyntaxiques. Ces classes grammaticales permettent de catégoriser les mots.

La CDGFr (grammaire de dépendances du français) compte 185 classes grammaticales permettant de catégoriser les mots du français. Ces classes grammaticales sont composées d'un nom de classe général (équivalant à une étiquette *POS*[2] : partie du discours) suivi de différentes informations grammaticales entre parenthèses. Par exemple un nom propre est catégorisé dans la classe *N(Lex=proper)* où *N* est la classe générale comprenant d'autres classes pour les noms telles que *N(Lex=common)* pour les noms communs, *N(Lex=time)* pour les noms faisant référence à des dates/périodes (e.g. « lundi », « midi », « janvier »), etc.

Dans la section 5.2 de nos travaux, nous utilisons deux sous-classifications de ces classes grammaticales : les classes grammaticales générales, qui conservent uniquement la partie du discours de chaque classe et les classes grammaticales étendues, qui incluent certaines informations des classes grammaticales de base. La seconde classification permet de disposer d'un jeu d'étiquettes étendu plus large que le jeu des classes générales mais plus restreint que l'ensemble de toutes les classes grammaticales. Le jeu des classes générales comporte 28 étiquettes et le jeu des classes étendues comporte 86 étiquettes. Une brève description des classes grammaticales générales est introduite dans la sous-section 4.4.1 du chapitre 4. Nous présentons ci-dessous la correspondance entre les classes grammaticales générales et les classes grammaticales étendues.

[1] Voir la section 4.3 du chapitre 4.
[2] *Part-Of-Speech*.

Classes générales	**Classes étendues**			
Adj	Adj Adjcompar	Adjquantifier	Adjsuperlative	Adjrestrictive
Adv	Adv Advcomme	Advquerestr	Advnotquerestr	Advapproxdegrcompar
N	N	Ntime	Ngener-quant	
Det	Det			
PN	PN PNrefl PNpersa PNqurel PNpersn PNindef	PNdema PNclita PNselect PNclitd PNclitl|d PNsubord	PNperscoref PNclit-copul PNclitg|p PNnotqurel PNdemgdol	PNattachnpersg|p PNattachpersa PNattachpersd PNattachnpersl|d PNattachpersn
PP	PP PPo PPd	PPgp PPl	PPagent PPinf	PPcirc|att PPselect
Vi	Vi	Vipzpres	Vifin	Viinf
Vt	Vtinf	Vtfin	Vtpzpres	
V2t	V2tfin	V2tinf	V2tpzpres	
Vaux	Vauxavoir	Vauxetre		
Vlight	Vlightfin	Vlightinf	Vlightpzpres	
Vcopul	Vcopulinf	Vcopulfin	Vcopulpzpres	
Conj	Conj Conjaggr	Conjcomp Interjection	Conjqucomp	Conjclauscond
Part	Parten	Partneg	Partemphat	
Num	Num	Numnotnum		
Colloc	Colloc			
Expletives	Expletives			
FullStop	FullStop			
QuestMark	QuestMark			
EmphatMark	EmphatMark			
Colon	Colon			
Dash	Dash			
SemiColon	SemiColon			
Quotes	Quotes			
Chevrons	Chevrons			
Parentheses	Parentheses			
Comma	Comma			
UT	UT			

C

Valeurs variables du paramètre d'élagage α pour la pré-annotation syntaxique

Lors des expérimentations d'analyse en dépendances (sous-section 5.4.4) de la section 5.4 (mise en place d'un processus complet de pré-annotation) du chapitre 5, le paramètre α est employé pour restreindre le nombre d'étiquettes syntaxiques (les noms des dépendances arrivant sur les mots) fournies en entrée de l'analyseur. La valeur de ce paramètre est utilisé pour exclure les étiquettes les moins probables de la liste d'étiquettes prédites pour chaque mot lors de l'étape d'étiquetage syntaxique en amont de l'analyse en dépendances (pour plus de détails sur l'élagage des listes, voir la section 5.4.3).

Les premières expérimentations d'analyse en dépendances présentées dans la sous-section 5.4.4 consistent à effectuer les analyses en utilisant un α fixe pour l'ensemble des phrases du corpus. Dans cette annexe, nous détaillons les valeurs de α utilisées lors de la dernière expérimentation présentée dans cette sous-section. La dernière expérimentation consiste à effectuer l'analyse en dépendances en utilisant un α variable en fonction de la longueur de la phrase traitée. Les valeurs de α utilisées pour cette expérimentation sont présentés dans le tableau ci-dessous, en fonction de la longueur des phrases traitées. On y reporte également le nombre moyen d'étiquettes syntaxiques conservées par mot pour chaque valeur de α, ainsi que la précision globale observée sur les listes d'étiquettes (le pourcentage de mots pour lesquels l'étiquette correcte appartient à la liste élaguée).

Longueur des phrases (nb mots)	α	**Nb moyen d'étiquettes par mot**	**Précision globale**
[0,10[	0,006	2,66	97,14
[10,15[	0,009	2,39	96,10
[15,20[	0,014	2,13	96,10
[20,30[	0,02	1,95	95,54
[30,35[	0,08	1,45	92,96
[35,40[	0,16	1,29	91,20
[40,45[	0,5	1,09	87,60
[45,∞[	0,9	1,01	85,22

D

Algorithme de la fonction Valence

La fonction **Valence**, présentée par l'algorithme 6, est utilisée par l'algorithme de conversion d'une structure de dépendances en séquence de transitions (6.3.1). Elle retourne la valence correspondante à la dépendance et à l'unité lexicale donnée en entrée :

- $\nwarrow l$ pour une dépendance sortante gauche étiquetée l ;
- $\nearrow l$ pour une dépendance sortante droite étiquetée l ;
- $\swarrow l$ pour une dépendance entrante gauche étiquetée l ;
- $\searrow l$ pour une dépendance entrante droite étiquetée l ;

Algorithme 6 : Valence

Données : $d = (i,l_k,t,j)$ une dépendance,
k l'indice du mot courant

```
si k = i alors
    si i > j alors
        retourner ↖l
    sinon
        retourner ↗l
sinon si k = j alors
    si i > j alors
        retourner ↙l
    sinon
        retourner ↘l
```

Bibliographie

ABEILLÉ, A., éditeur (2003). *Treebanks : Building and Using Parsed Corpora*. Springer. 151

ABEILLÉ, A., CLÉMENT, L. et TOUSSENEL, F. (2003). *Building a Treebank for French*, pages 168–188. *In* Abeillé (2003). 39, 94, 132

AHO, A. V. et ULLMAN, J. D. (1972). *The Theory of Parsing, Translation, and Compiling*. Prentice-Hall, Inc. 55

AJDUKIEWICZ, K. (1935). Die syntaktische konnexität. 61

ATTARDI, G. (2006). Experiments with a multilanguage non-projective dependency parser. *In Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 166–170, New York City, United States. 38, 57

BAKER, J. K. (2005). Trainable Grammars for Speech Recognition. *The Journal of the Acoustical Society of America*, 65:132. 55

BALLESTEROS, M. et NIVRE, J. (2012). MaltOptimizer : A System for MaltParser Optimization. *In Proceedings of the Eighth International Conference on Language Resources and Evaluation*, LREC 2012, pages 2757–2763. 96, 130

BANGALORE, S. et JOSHI, A., éditeurs (2010). *Complexity of Lexical Descriptions and its Relevance to Natural Language Processing : A Supertagging Approach*. MIT Press. 152, 155, 157, 159

BANGALORE, S. et JOSHI, A. K. (1999). Supertagging : An Approach to Almost Parsing. *Computational Linguistics*, 25:237–265. 53, 54

BAR-HILLEL, Y., GAIFMAN, C. et SHAMIR, E. (1964). On Categorial and Phrase Structure Grammars. *Bulletin of the Research Council of Israel*, pages 99–115. 33, 61

BAUM, L. E. (1972). An Equality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of Markov Processes. *Inequalities*, 3:1–8. 52

BÉCHET, D., DIKOVSKY, A. et LACROIX, O. (2014). "CDG Lab" : an Integrated Environment for Categorial Dependency Grammar and Dependency Treebank Development. *In* Gerdes *et al.* (2014), pages 153–169. 16, 17, 69, 139

BOGUSLAVSKY, I., GRIGORIEVA, S., GRIGORIEV, N., KREIDLIN, L. et FRID, N. (2000). Dependency Treebank for Russian : Concept, Tools, Types of Information. *In the 18th International Conference on Computational Linguistics*, COLING 2000, Saarbrüken. 43

BOGUSLAVSKY, I., IOMDIN, L., TIMOSHENKO, S. et FROLOVA, T. (2009). Development of the Russian Tagged Corpus with Lexical and Functional Annotation. *In MONDILEX Third Open Workshop 2009*, Bratislava. 43

BOHNET, B. (2010). Very High Accuracy and Fast Dependency Parsing is Not a Contradiction. *In Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, Beijing, China. 35, 96

BOUDIN, F. et HERNANDEZ, N. (2012). Détection et correction automatique d'erreurs d'annotation morpho-syntaxique du french treebank (detecting and correcting pos annotation in the french treebank) [in french]. *In Actes de la conférence conjointe JEP-TALN-RECITAL 2012, volume 2 : TALN*, Grenoble, France. 51

BOULLIER, P. (2010). *Supertagging : a non-statistical parsing-based approach*. *In* Bangalore et Joshi (2010). 54

BRANTS, S., DIPPER, S., HANSEN, S., LEZIUS, W. et SMITH, G. (2002). The TIGER Treebank. *In Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol, Bulgaria. 43

BRANTS, T. (2000). TnT - A Statistical Part-of-Speech Tagger. *In Proceedings of the 6th Applied Natural Language Processing Conference*, ANLP 2000, Seattle. 52

BRILL, E. (1995). Transformation-Based Error-Driven Learning and Natural Language Processing : A Case Study in Part-of-Speech Tagging. *In* Dale (1995), pages 543–565. 51

BUCHHOLZ, S. et MARSI, E. (2006). CoNLL-X Shared Task on Multilingual Dependency Parsing. *In Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X, pages 149–164, New York City. 44

CANDITO, M., CRABBÉ, B., DENIS, P. et GUÉRIN, F. (2009). Analyse syntaxique du français : des constituants aux dépendances. *In Actes de la 16ème conférence sur le Traitement Automatique des Langues Naturelles*, TALN 2009, Senlis, France. 39, 41, 42

CANDITO, M., PERRIER, G., GUILLAUME, B., RIBEYRE, C., FORT, K., SEDDAH, D. et de la CLERGERIE, E. (2014). Deep Syntax Annotation of the Sequoia French Treebank. *In Proceedings of the Ninth International Conference on Language Resources and Evaluation*, LREC 2014, pages 2298–2305, Reykjavik, Iceland. ACL Anthology Identifier : L14-1410. 41

CANDITO, M. et SEDDAH, D. (2012a). Le corpus Sequoia : annotation syntaxique et exploitation pour l'adaptation d'analyseur par pont lexical. *In Actes de la conférence conjointe JEP-TALN-RECITAL 2012, volume 2 : TALN*, Grenoble, France. 39, 102

CANDITO, M. et SEDDAH, D. (2012b). Effectively long-distance dependencies in French : annotation and parsing evaluation. *In Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, TLT 11, Lisbon, Portugal. 42

CERISARA, C., GARDENT, C. et ANDERSON, C. (2010). Building and Exploiting a Dependency Treebank for French Radio Broadcast. *In Proceedings of the Ninth International Workshop on Treebanks and Linguistic Theories*, TLT9, Tartu, Estonia. 43

CHARNIAK, E. et JOHNSON, M. (2005). Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. *In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL 2005, pages 173–180. 56

CHEN, J. (2001). *Towards Efficient Statistical Parsing using Lexicalized Grammatical Information*. Thèse de doctorat. 54

CHENG, Y., ASAHARA, M. et MATSUMOTO, Y. (2005a). Chinese deterministic dependency analyzer : Examining effects of global features and root node finder. *In Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, pages 17–24. 114

CHENG, Y., ASAHARA, M. et MATSUMOTO, Y. (2005b). Machine Learning-based Dependency Analyzer for Chinese. *Journal of Chinese Language and Computing*, 15(1):13–24. 57

CHOI, J. D. et PALMER, M. (2011). Getting the Most out of Transition-based Dependency Parsing. *In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies*, ACL 2011, Portland. 38

CHOMSKY, N. (1959). On certain formal properties of grammars. *Information and control*, 2(2): 137–167. 30

CHU, Y.-J. et LIU, T.-H. (1965). On the shortest arborescence of a directed graph. *In Science Sinica*, volume 14, pages 1396–1400. 35

CLARK, S. et CURRAN, J. R. (2004). The Importance of Supertagging for Wide-Coverage CCG Parsing. *In Proceedings of the 20th International Conference on Computational Linguistics*, COLING'04, pages 282–288, Genève, Suisse. 54

CLARK, S. et CURRAN, J. R. (2007). Wide-Coverage Efficient Statistical Parsing with CCG and Log-Linear Models. *Computational Linguistics*, 33(4):493–552. 56, 141

COLLINS, M. (2002). Discriminative training methods for hidden markov models : Theory and experiments with perceptron algorithms. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP 2002, Philadelphia, USA. 53

COLLINS, M. (2003). Head-Driven Statistical Models for Natural Language Parsing. *Computational linguistics*, 29(4):589–637. 55

COLLINS, M. et KOO, T. (2005). Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1):25–70. 56

CONSTANT, M., TELLIER, I., DUCHIER, D., DUPONT, Y., SIGOGNE, A. et BILLOT, S. (2011). Intégrer des connaissances linguistiques dans un CRF : application à l'apprentissage d'un segmenteur-étiqueteur du français. *In Actes de la 18ème conférence sur le Traitement Automatique des Langues Naturelles*, TALN 2011, Montpellier, France. 52

COURTOIS, B. (1990). Un système de dictionnaires électroniques pour les mots simples du français. *In Langue Française*, volume 87, pages 11–22. Armand Colin. 52

COVINGTON, M. A. (2001). A fundamental algorithm for dependency parsing. *In Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102. 38, 130

CRABBÉ, B. et CANDITO, M. (2008). Expériences d'analyse syntaxique statistique du français. *In Actes de la 15ème conférence sur le Traitement Automatique des Langues*, TALN 2008, Avignon, France. 50, 51, 94

CRAMMER, K. et SINGER, Y. (2003). Ultraconservative Online Algorithms for Multiclass Problems. *The Journal of Machine Learning Research*, 3:951–991. 57

DALE, R., éditeur (1995). *Computational Linguistics*. Volume 21 (4). Massachusetts Institute of Technology. 152

DAS, D. et PETROV, S. (2011). Unsupervised Part-of-Speech Tagging with Bilingual Graph-Based Projections. *In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics : Human Language Technologies*, ACL'11, pages 600–609, Portland, Oregon, USA. 42

de MARNEFFE, M.-C., MACCARTNEY, B. et MANNING, C. D. (2006). Generating Typed Dependency Parses from Phrase Structure Parses. *In Proceedings of the fifth international conference on Language Resources and Evaluation*, LREC'06, Genoa, Italy. 44

de MARNEFFE, M.-C. et MANNING, C. D. (2008). The Stanford Typed Dependencies Representation. *In Proceedings of the Coling 2008 Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, Manchester, United Kingdom. 29, 42

DEBUSMANN, R., DUCHIER, D. et KRUIJFF, G.-J. M. (2004). Extensible Dependency Grammar : A New Methodology. *In Proceedings of the COLING 2004 Workshop on Recent Advances in Dependency Grammar*, pages 70–76. 32

DEKHTYAR, M. et DIKOVSKY, A. (2004). Categorial Dependency Grammars. *In Proceedings of the 7th International Workshop on Categorial Grammars*, CG'04, pages 76–91, Montpellier, France. 16, 28, 33, 59, 139

DEKHTYAR, M. et DIKOVSKY, A. (2008). Generalized Categorial Dependency Grammars. LNCS 4800, pages 230–255. Springer. 16, 59, 62, 139

DENIS, P. et SAGOT, B. (2009). Coupling an Annotated Corpus and a Morphosyntactic Lexicon for State-of-the-Art POS Tagging with Less Human Effort. *In Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*, PACLING 2009, Hong Kong, China. 51, 52, 94, 96, 119

DENIS, P. et SAGOT, B. (2012). *Coupling an annotated corpus and a lexicon for state-of-the-art POS tagging*, volume 46, pages 721–736. 51, 52, 94, 96, 119

DIKOVSKY, A. (2004). Dependencies as Categories. *In Proceedings of the Coling 2004 Workshop "Recent Advances in Dependency Grammars"*, pages 90–97, Genève, Suisse. 62

DIKOVSKY, A. (2011). Categorial Dependency Grammars : from Theory to Large Scale Grammars. *In Proceedings of the International Conference on Dependency Linguistics*, DEPLING 2011, Barcelona, Spain. 16, 33, 60, 67, 139

DUCHIER, D. et DEBUSMANN, R. (2001). Topological Dependency Trees : A Constraint-Based Account of Linear Precedence. *In Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 180–187. 32

EARLEY, J. (1970). An efficient context-free parsing algorithm. *Communications of the ACM*, 13:94–102. 30

EDMONDS, J. (1967). Optimum Branchings. *Journal of Research of the National Bureau of Standards*, 71B:233–240. 35

EISNER, J. (1996). Three New Probabilistic Models for Dependency Parsing : An Exploration. *In Proceedings of the 16th conference on Computational linguistics*, COLING 1996, pages 340–345, Copenhagen, Denmark. 30

EISNER, J. (1997). Bilexical Grammars and a Cubic-Time Probabilistic Parser. *In Proceedings of the Fifth International Workshop on Parsing Technologies*. 56

FRANCIS, W. et KUCERA, H. (1982). Frequency analysis of English usage. 51

GAIFMAN, H. (1965). Dependency systems and phrase-structure systems. *Information and Control*, 8:304–337. 32

GERDES, K., HAJIČOVÁ, E. et WANNER, L., éditeurs (2014). volume 258 de *Frontiers in Artificial Intelligence and Applications*. IOS Press. 151

GIMÉNEZ, J. et MÁRQUEZ, L. (2004). SVMTool : A general POS tagger generator based on Support Vector Machines. *In Proceedings of the 4th International Conference on Language Resources and Evaluation*, LREC 2004, Lisbon, Portugal. 53

GREENE, B. B. et RUBIN, G. M. (1971). Automatic grammatical tagging of English. Rapport technique, Department of Linguistics, Brown University. 51

HAJIČ, J., CIARAMITA, M., JOHANSSON, R., KAWAHARA, D., MARTÍ, M. A., MÀRQUEZ, L., MEYERS, A., NIVRE, J., PADÓ, S., ŠTĚPÁNEK, J., STRAŇÁK, P., SURDEANU, M., XUE, N. et ZHANG, Y. (2009). The CoNLL-2009 Shared Task : Syntactic and Semantic Dependencies in Multiple Languages. *In Proceedings of the Thirteenth Conference on Computational Natural Language Learning : Shared Task*, pages 1–18, Boulder, Colorado. 44

HAJIČ, J. et ZEMÁNEK, P. (2004). Prague Arabic dependency treebank : Development in data and tools. *In In Proceedings of the International Conference on Arabic Language Resources and Tools*, NEMLAR, pages 110–117. 43

HAJIČOVÁ, E., HAJIČ, J., HOLUB, M., PAJAS, P., KOLÁŘOVÁ-ŘEZNÍČKOVÁ, V., SGALL, P. et HLADKÁ, B. V. (2001). The Current Status of the Prague Dependency Treebank. *In Proceedings of the 4th International Conference on Text, Speech and Dialogue*, TSD 2001, Železná Ruda-Špičák, Czech Republic. 43

HALL, J. (2006). Discriminative Classifiers for Deterministic Dependency Parsing. *In Proceedings of the 44th Annual Meeting of the Association for ComputationalLinguistics*, ACL'06. 114

HALL, K. et NOVÁK, V. (2005). Corrective Modeling for Non-projective Dependency Parsing. *In Proceedings of the Ninth International Workshop on Parsing Technology*, IWPT 2005, pages 42–52. 124

HARBUSCH, K., BÄCKER, J. et HASAN, S. (2010). Applications of HMM-Based Supertagging. *In* Bangalore et Joshi (2010), page 449. 54

HARPER, M. P. et HELZERMAN, R. A. (1995). Extensions to Constraint Dependency Parsing for Spoken Language Processing. *Computer Speech and Language*, 9:187–234. 35

HAYS, D. G. (1965). Dependency Theory : A Formalism and Some Observations. *Language*, 40(4): 511–525. 32

HUANG, L. (2008). Forest reranking : Discriminative parsing with non-local features. *In Proceedings of the 46th Annual Meeting of Association for Computational Linguistics : Human Language Technologies*, ACL'08, pages 586–594. 56

HUANG, L. et SAGAE, K. (2010). Dynamic Programming for Linear-Time Incremental Parsing. *In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL'10, Uppsala, Sweden. 38

HUDSON, R. (1990). *English Word Grammar*. Wiley-Blackwell. 32, 33

JOSHI, A. et RAMBOW, O. (2003). A Formalism for Dependency Grammar Based on Tree Adjoining Grammar. *In Proceedings of the Conference on Meaning-Text Theory*, MTT 2003, pages 207–216. 32

JOSHI, A. K., LEVY, L. S. et TAKAHASHI, M. (1975). Tree Adjunct Grammars. *Journal of Computer and System Sciences*, 10:136–163. 32

JURAFSKY, D. et MARTIN, J. (2000). *Speech and Language Processing : An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall series in artificial intelligence. Prentice Hall. 31, 55

KAHANE, S. (1997). Bubble Trees and Syntactic Representations. *In Proceedings of the 5th Meeting the Mathematics of Language*, MOL5, Saarbrücken. 29

KAPLAN, R. M. et BRESNAN, J. (1995). Lexical-functional grammar : A formal system for grammatical representation. *Formal Issues in Lexical Functional Grammar*, pages 173–281. 32

KARLOV, B. et LACROIX, O. (2012). Prémices d'une analyse syntaxique par transition pour des structures de dépendance non-projectives. *In Actes de la conférence conjointe JEP-TALN-RECITAL 2012 : volume 3*, Grenoble. 17, 108

KING, T. H., CROUCH, R., RIEZLER, S., DALRYMPLE, M. et KAPLAN, R. M. (2003). The PARC 700 Dependency Bank. *In Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, EACL 2003, Budapest. 44

KUDO, T. et MATSUMOTO, Y. (2002). Japanese dependency analysis using cascaded chunking. *In Proceedings of the 6th conference on Natural language learning*, CoNLL 2002, Taipei, Taiwan. 37, 57

KÜBLER, S., MCDONALD, R. et NIVRE, J. (2009). *Dependency parsing*. Morgan et Claypool. 32, 34, 35

LACHERET, A., KAHANE, S., BELIAO, J., DISTER, A., GERDES, K., GOLDMAN, J.-P., OBIN, N., PIETRANDREA, P. et TCHOBANOV, A. (2014). Rhapsodie : a prosodic-syntactic treebank for spoken french. *In Proceedings of the Ninth International Conference on Language Resources and Evaluation*, LREC'14, pages 295–301, Reykjavik, Iceland. 43

LACROIX, O. (2013a). Influence de l'étiquetage syntaxique des têtes sur l'analyse en dépendances discontinues du français. *In Actes de TALN-RÉCITAL 2013*, volume 2, pages 110–123, Les Sables d'Olonne, France. 17

LACROIX, O. (2013b). On the Effect of Head Tagging on Parsing Discontinuous Dependencies in French. *In Proceedings of the ESSLLI Student Session 2013*, pages 93–103, Düsseldorf, Allemagne. 17

LACROIX, O. et BÉCHET, D. (2014a). A three-step transition-based system for non-projective dependency parsing. *In Proceedings of the 25th International Conference on Computational Linguistics*, Coling 2014, Dublin, Irlande. 17

LACROIX, O. et BÉCHET, D. (2014b). Validation issues induced by an automatic pre-annotation mechanism in the building of non-projective dependency treebanks. *In Proceedings of the Ninth International Conference on Language Resources and Evaluation*, LREC'14, Reykjavik, Islande. 17

LACROIX, O., BÉCHET, D. et BOUDIN, F. (2014). Label pre-annotation for building non-projective dependency treebanks for french. *In Proceedings of the 15th International Conference on Intelligent Text Processing and Computational Linguistics*, Kathmandu, Népal. 17

LAFFERTY, J., MCCALLUM, A. et PEREIRA, F. (2001). Conditional Random Fields : Probabilistic Models for Segmenting and Labeling Sequence Data. *In Proceedings of the 28th International Conference on Machine Learning*, ICML 2001, Williamstown. 49

LAFFERTY, J., SLEATOR, D. et TEMPERLEY, D. (1992). *Grammatical trigrams : A probabilistic model of link grammar*. School of Computer Science, Carnegie Mellon University. 56

LAVERGNE, T., CAPPÉ, O. et YVON, F. (2010). Practical very large scale CRFs. *In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL'10, Uppsala, Suède. 79, 87

LIU, H. et HUANG, W. (2006). A Chinese Dependency Syntax for Treebanking. *In Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation*, PACLIC'06, China. 27

MAGERMAN, D. M. (1995). Statistical Decision-Tree Models for Parsing. *In Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, ACL'95, pages 276–283, Cambridge, USA. 41

MARCUS, M. P., SANTORINI, B. et MARCINKIEWICZ, M. A. (1993). Building a Large Annotated Corpus of English : The Penn Treebank. *Computational Linguistics*, 19(2):313–330. 44

MARUYAMA, H. (1990). Structural Disambiguation with Constraint Propagation. *In Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics*, ACL 1990, Pittsburgh. 34

MCCALLUM, A., FREITAG, D. et PEREIRA, F. C. (2000). Maximum Entropy Markov Models for Information Extraction and Segmentation. *In Proceedings of the Seventeenth International Conference on Machine Learning*, ICML 2000, pages 591–598, Stanford, CA, USA. 49

MCDONALD, R., CRAMMER, K. et PEREIRA, F. (2005). Online large-margin training of dependency parsers. *In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, ACL'05, Ann Arbor, USA. 35, 57

MCDONALD, R., NIVRE, J., QUIRMBACH-BRUNDAGE, Y., GOLDBERG, Y., DAS, D., GANCHEV, K., HALL, K., PETROV, S., ZHANG, H., TÄCKSTRÖM, O., BEDINI, C., BERTOMEU CASTELLÓ, N. et LEE, J. (2013). Universal Dependency Annotation for Multilingual Parsing. *In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics : volume 2*, ACL'13, Sofia, Bulgaria. 42, 43, 132, 137, 141

MCDONALD, R. et PEREIRA, F. (2006). Online Learning of Approximate Dependency Parsing Algorithms. *In Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, EACL 2006, Trento, Italy. 36, 124

MCDONALD, R., PEREIRA, F., RIBAROV, K. et HAJIC, J. (2005). Non-projective Dependency Parsing using Spanning Tree Algorithms. *In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT/EMNLP 2005, Vancouver. 35

MCDONALD, R., PETROV, S. et HALL, K. (2011). Multi-Source Transfer of Delexicalized Dependency Parsers. *In Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP'11, pages 62–72, Edinburgh, Scotland, UK. 42

MEL'CUK, I. (1988). *Dependency syntax : Theory and Practice*. State University of New York Press. 11, 15, 23, 24, 43, 59

MOOT, R. (2010). *Extraction of Type-Logical Supertags from the Spoken Dutch Corpus*. *In* Bangalore et Joshi (2010). 54

NASR, A. et RAMBOW, O. (2010). *Non-lexical chart parsing for TAG*. *In* Bangalore et Joshi (2010). 54

NIVRE, J. (2003). An Efficient Algorithm for Projective Dependency Parsing. *In Proceedings of the 8th International Workshop on Parsing Technologies*, IWPT 2003, Nancy, France. 38, 116, 125

NIVRE, J. (2006). *Inductive Dependency Parsing*. Springer Netherlands. 38

NIVRE, J. (2008). Algorithms for Deterministic Incremental Dependency Parsing. *Computational Linguistics*, 34(4):513–553. 36, 111

NIVRE, J. (2010). *In* INDURKHYA, N. et DAMERAU, F. J., éditeurs : *Handbook of Natural Language Processing*, chapitre Statistical Parsing. CRC Press. 57

NIVRE, J., HALL, J. et NILSSON, J. (2004). Memory-Based Dependency Parsing. *In Proceedings of the Eighth Conference on Computational Natural Language Learning*, CoNLL'04, pages 49–56. 57

NIVRE, J., HALL, J. et NILSSON, J. (2006a). MaltParser : A Data-Driven Parser-Generator for Dependency Parsing. *In Proceedings of the Fifth International Conference on Language Resources and Evaluation*, LREC'06, Genoa. 38, 96, 111, 125, 130, 140

NIVRE, J., HALL, J., NILSSON, J., ERYIGIT, G. et MARINOV, S. (2006b). Labeled Pseudo-Projective Dependency Parsing with Support Vector Machines. *In Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X, pages 221–225. Association for Computational Linguistics. 57

NIVRE, J. et NILSSON, J. (2005). Pseudo-projective Dependency Parsing. *In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL'05, pages 99–106, Ann Arbor, Michigan. 38, 124, 130, 137, 141

PETROV, S., DAS, D. et MCDONALD, R. (2012). A Universal Part-of-Speech Tagset. *In Proceedings of the Eighth International Conference on Language Resources and Evaluation*, LREC'12, Istanbul, Turkey. 42

POLLARD, C. et SAG, I. A. (1987). *Information-Based Syntax and Semantics, Vol. 1 : Fundamentals*. Cambridge University Press. 32

RABINER, L. R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *In Proceedings of IEEE 1989*. 49

RATNAPARKHI, A. (1996). A Maximum Entropy Model for Part-Of-Speech Tagging. *In Proceesings of the first Conference on Empirical Methods in Natural Language Processing*, EMNLP 1996, University of Pennsylvania. 49, 87

RESNIK, P. (1992). Probabilistic Tree-Adjoining Grammar as a Framework for Statistical Natural Language Processing. *In Proceedings of the 15th International Conference on Computational Linguistics*, volume 2 de *Coling 1992*, pages 418–424. 56

ROSENBLATT, F. (1958). The Perceptron : A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, pages 386–408. 49

ROTH, D. (1998). Learning to resolve natural language ambiguities : A unified approach. *In Proceedings of the Fifteenth National Conference on Artificial Intelligence*, AAAI-98, pages 806–813. 54

SAGOT, B. (2010). The Lefff, a freely available and large-coverage morphological and syntactic lexicon for French. *In Proceedings of the Seventh conference on International Language Resources and Evaluation*, LREC'10, Valletta, Malte. 52, 67

SARKAR, A. (2010). *Combining Supertagging and Lexicalized Tree-Adjoining Grammar Parsing*. *In* Bangalore et Joshi (2010). 54

SCHABES, Y. (1990). *Mathematical and computational aspects of lexicalized grammars*. Thèse de doctorat. 32, 53

SCHABES, Y. (1992). Stochastic Lexicalized Tree-Adjoining Grammars. *In Proceedings of the 15th International Conference on Computational Linguistics*, volume 2 de *COLING 1992*, pages 425–432. 56

SCHMID (1994). Probabilistic Part-of-Speech Tagging Using Decision Trees. *In Proceedings of the international conference on new methods in language processing*, NMLP 1994, Manchester. 52

SCHRÖDER, I., POP, H. F., MENZEL, W. et FOTH, K. A. (2001). Learning Grammar Weights Using Genetic Algorithms. *In Proceedings of the Conference "Recent Advances in Natural Language Processing"*, RANLP 2001, Tzigov Chark, Bulgaria. 35

SEDDAH, D., CANDITO, M. et ANGUIANO, E. H. (2013a). A word clustering approach to domain adaptation : Robust parsing of source and target domains. *Journal of Logic and Computation*. 141

SEDDAH, D., TSARFATY, R., KÜBLER, S., CANDITO, M., CHOI, J. D., FARKAS, R., FOSTER, J., GOENAGA, I., GALLETEBEITIA, K. G., GOLDBERG, Y., GREEN, S., HABASH, N., KUHLMANN, M., MAIER, W., MARTON, Y., NIVRE, J., PRZEPIÓRKOWSKI, A., ROTH, R., SEEKER, W., VERSLEY, Y., VINCZE, V., WOLIŃSKI, M. et WRÓBLEWSKA, A. (2013b). Overview of the SPMRL 2013 Shared Task : A Cross-Framework Evaluation of Parsing Morphologically Rich Languages. *In Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 146–182, Seattle, WA, USA. 44

SHEN, L. (2010). Discriminative Learning of Supertagging. *In* Bangalore et Joshi (2010), page 159. 54

SLEATOR, D. D. et TEMPERLEY, D. (1995). Parsing English with a Link Grammar. *Computing Research Repository*, abs/cmp-lg. 32, 33

STOLCKE, A. (1995). An Efficient Probabilistic Context-Free Parsing Algorithm that Computes Prefix Probabilities. *Computational linguistics*, 21(2):165–201. 55

TESNIÈRE, L. (1959). *Éléments de syntaxe structurale*. Klincksieck. 15, 23, 28, 59

TOUTANOVA, K. et MANNING, C. D. (2000). Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. *In Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*, pages 63–70. 53

URIELI, A. (2013). *Analyse syntaxique robuste du français : concilier méthodes statistiques et connaissances linguistiques dans l'outil Talismane*. Thèse de doctorat, Université Toulouse le Mirail-Toulouse II. 58, 111

VAPNIK, V. N. (1998). *Statistical Learning Theory*. Wiley. 49

VITERBI, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE*, 13(2):260–269. 52

WANG, W. et HARPER, M. P. (2004). A Statistical Constraint Dependency Grammar (CDG) Parser. *In Proceedings of the Workshop on Incremental Parsing : Bringing Engineering and Cognition Together*, IncrementParsing '04, pages 42–49, Barcelona, Spain. 56

YAMADA, H. et MATSUMOTO, Y. (2003). Statistical Dependency Analysis with Support Vector Machines. *In Proceedings of the 8th International Workshop on Parsing Technologies*, IWPT 2003, Nancy. 37, 57, 108, 114

YOUNGER, D. H. (1967). Recognition and Parsing of Context-Free Languages in Time n3. *Information and Control*, 10:189–208. 30

ZHANG, H. et MCDONALD, R. (2012). Generalized Higher-Order Dependency Parsing with Cube Pruning. *In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CONLL 2012, pages 320–331, Jeju Island, Korea. 36

ZHANG, Y. et CLARK, S. (2008). A Tale of Two Parsers : Investigating and Combining Graph-based and Transition-based Dependency Parsing Using Beam-search. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 562–571, Honolulu, Hawaii. 58

ZHANG, Y. et CLARK, S. (2011). Syntactic Processing Using the Generalized Perceptron and Beam Search. *Computational Linguistics*, 37(1):105–151. 58

l'université nantes angers le mans
COMMUNAUTÉ D'UNIVERSITÉS ET ÉTABLISSEMENTS

UNIVERSITÉ DE NANTES

Thèse de Doctorat

Ophélie LACROIX

De l'étiquetage syntaxique pour les grammaires catégorielles de dépendances à l'analyse par transition dans le domaine de l'analyse en dépendances non-projective

From syntactic tagging for categorial dependency grammars to transition-based parsing in the domain of non-projective dependency parsing

Résumé
Cette thèse prend place dans le domaine de l'analyse syntaxique en dépendances. D'une part nous étudions l'impact d'une méthode statistique d'étiquetage syntaxique sur un analyseur basé sur les grammaires catégorielles de dépendances. Nous proposons en ce sens un processus complet de pré-annotation comprenant la segmentation des phrases en mots (incluant les mots composés), l'étiquetage grammatical et syntaxique de ces mots et l'analyse en dépendances de la phrase dans le but d'alléger le travail des annotateurs dans le cadre de la construction de corpus en dépendances non-projectifs pour le français.
D'autre part, nous étudions également les méthodes intégralement dirigées par les données dans le domaine de l'analyse en dépendances à travers l'adaptation d'un analyseur par transition à la représentation en dépendances des grammaires catégorielles de dépendances. Puis nous proposons une méthode séparant les étapes de prédiction des dépendances projectives et non-projectives dans le but d'améliorer la prédiction des dépendances non-projectives. Nous montrons que cette méthode est adaptable à n'importe quel corpus en dépendances standard.

Mots clés
analyse syntaxique en dépendances, grammaires catégorielles de dépendances, étiquetage syntaxique, analyse par transition.

Abstract
This thesis takes place in the domain of syntactic dependency parsing. On the one hand we study the effect of a statistical method for syntactic tagging on a CDG-based parser (Categorial Dependency Grammar). We propose a pre-annotation process which includes the word-segmentation of sentences, the POS-tagging and the syntactic tagging of those words and the dependency analysis in order to alleviate the burden of the annotators in the context of the building of non-projective dependency treebanks for French.
On the other hand, we study a data-driven method for dependency parsing through the adaptation of a transition-based parser to the dependency representation induced by the categorial dependency grammars. Moreover, we propose a three-steps transition-based method which performs separatly the prediction of the projective dependencies first and then the right and left non-projective dependencies in order to increase the prediction scores on non-projective dependencies. We show this method can be adapted to any standard dependency treebank.

Key Words
dependency parsing, categorial dependency grammar, syntactic tagging, transition-based parsing.