Grammaires locales pour l'analyse automatique de textes : méthodes de construction et outils de gestion

Mathieu Constant

Université de Marne-La-Vallée

Institut Gaspard Monge

Ecole Doctorale Information, Communication, Modélisation et Simulation

Institut Gaspard Monge, laboratoire d'informatique

THÈSE

pour obtenir le grade de

Docteur de l'Université de Marne-La-Vallée

Discipline : informatique linguistique

présentée et soutenue publiquement par

Matthieu CONSTANT

le 8 septembre 2003

Grammaires locales pour l'analyse automatique de textes : méthodes de construction et outils de gestion

Local grammars for text parsing: construction methods and management tools

Directeurs de thèse

†Maurice GROSS

Éric LAPORTE

Jury : Maxime CROCHEMORE
Jacqueline GIRY-SCHNEIDER
Franz GUENTHNER (*Rapporteur*)
Éric LAPORTE
Denis MAUREL (*Rapporteur*)

Remerciements

La thèse présentée dans cet ouvrage n'aurait jamais vu le jour sans ma rencontre avec Maurice Gross à qui je souhaite rendre hommage. J'espère que ce travail permettra d'apporter un témoignage supplémentaire de l'extraordinaire richesse de son œuvre scientifique.

Je souhaite exprimer tous mes remerciements aux membres du jury : tout d'abord, à mon directeur Eric Laporte qui a eu le courage de prendre la suite de Maurice Gross et dont les remarques, corrections et conseils ont apporté un nouvel éclairage à mon travail ; ensuite, aux rapporteurs, Franz Guenthner et Denis Maurel, pour leurs commentaires et remarques pertinentes ; enfin à Maxime Crochemore et Jacqueline Giry-Schneider pour leurs conseils enrichissants durant la soutenance.

Je voudrais spécialement remercier mon collègue et ami Takuya Nakamura pour son soutien permanent et les longues conversations linguistiques que nous avons eues ensemble. Je voudrais également exprimer ma gratitude à Sébastien Paumier, mon premier secours technique, pour sa précieuse collaboration scientifique.

Je tiens à exprimer toute ma reconnaissance à tous les membres de l'équipe d'informatique linguistique de Marne-la-Vallée dont le soutien et la gentillesse ont joué un rôle fondamental dans l'accomplissement de cette thèse. Je voudrais également remercier Cédrick Fairon et Max Silberztein pour leur remarques sur mon travail informatique. Je tiens à exprimer mes remerciements sincères à Anastasia Yannacopoulou pour sa collaboration et son soutien administratif. Je voudrais aussi dire un grand merci à Anne Dister pour sa collaboration dans l'événementiel. Merci également à l'équipe portugaise pour son accueil exceptionnel lors de mes séjours à Lisbonne.

Je ne pourrai terminer cette série de remerciements sans avoir une pensée pour tous ceux que je n'ai pas cité et qui n'ont pas besoin de l'être pour savoir qu'ils comptent.

Résumé

L'explosion du nombre de documents disponibles (notamment sur Internet) a rendu le domaine du Traitement Automatique des Langues (TAL) et ses outils incontournables. De nombreux chercheurs marquent l'importance de la linguistique dans ce domaine. Ils préconisent la construction de larges bases de descriptions linguistiques, composées de lexiques et de grammaires. Cette démarche a un gros inconvénient : elle nécessite un investissement lourd qui s'inscrit sur le long terme. Pour palier à ce problème, il est nécessaire de mettre au point des méthodes et des outils informatiques d'aide à la construction de composants linguistiques fins et directement applicables à des textes. Nous nous sommes penché sur le problème des grammaires locales qui décrivent des contraintes précises et locales sous la forme de graphes. Deux questions fondamentales se posent :

- Comment construire efficacement des grammaires précises, complètes et applicables à des textes ?
- Comment gérer leur nombre et leur éparpillement ?

Comme solution au premier problème, nous avons proposé un ensemble de méthodes simples et empiriques. Nous avons exposé des processus d'analyse linguistique et de représentation à travers deux phénomènes : les expressions de mesure (*un immeuble d'une hauteur de 20 mètres*) et les adverbes de lieu contenant un nom propre locatif (*à l'île de la Réunion*), deux points critiques du TAL. Sur la base de M. Gross (1975), nous avons ramené chaque phénomène à une phrase élémentaire. Ceci nous a permis de classer sémantiquement certains phénomènes au moyen de critères formels. Nous avons systématiquement étudié le comportement de ces phrases selon les valeurs lexicales de ses éléments. Les faits observés ont ensuite été représentés formellement soit directement dans des graphes à l'aide d'un éditeur, soit par l'intermédiaire de tables syntaxiques ensuite converties semi-automatiquement en graphes. Au cours de notre travail, nous avons été confronté à des systèmes relationnels de tables syntaxiques pour lesquels la méthode standard de conversion due à E. Roche (1993) ne fonctionnait plus. Nous avons donc élaboré une nouvelle méthode adaptée avec des formalismes et des algorithmes permettant de gérer le cas où les informations sur les graphes à construire se trouvent dans plusieurs tables.

En ce qui concerne le deuxième problème, nous avons proposé et implanté un prototype de système de gestion de grammaires locales : une bibliothèque en-ligne de graphes. Le but à terme est de centraliser et de diffuser les grammaires locales construites au sein du réseau RELEX. Nous avons conçu un ensemble d'outils permettant à la fois de stocker de nouveaux graphes et de rechercher des graphes suivant différents critères. L'implémentation d'un moteur de recherche de grammaires nous a également permis de nous pencher sur un nouveau champ d'investigation dans le domaine de la recherche d'information : la recherche d'informations linguistiques dans des grammaires locales.

Mots-clés : analyse automatique de textes, analyse syntaxique, complément prépositionnel locatif, conversion de tables syntaxiques en graphes, expression de mesure, gestion de grammaires locales, grammaire locale, graphe, lexique-grammaire, nom propre géographique, recherche automatique d'informations linguistiques, réseau récursif de transitions

Abstract

Many researchers in the field of Natural Language Processing have shown the significance of descriptive linguistics and especially the use of large-scaled databases of fine-grained linguistic components composed of lexicons and grammars. This approach has a drawback: it requires long-term investment. It is then necessary to develop methods and computational tools to help the construction of such data that are required to be directly applicable to texts. This work focuses on a specific linguistic representation: local grammars that describe precise and local constraints in the form of graphs. Two issues arise:

- How to efficiently build precise, complete and text-applicable grammars?
- How to deal with their growing number and their dispersion?

To handle the first problem, a set of simple and empirical methods have been exposed on the basis of M. Gross (1975)'s lexicon-grammar methodology. The whole process of linguistic analysis and formal representation has been described through the examples of two original phenomena: expressions of measurement (*un immeuble d'une hauteur de 20 mètres*) and locative prepositional phrases containing geographical proper names (*à l'île de la Réunion*). Each phenomenon has been narrowed to elementary sentences. This enables semantically classify them according to formal criteria. The syntactical behavior of these sentences has been systematically studied according to the lexical value of their elements. Then, the observed properties have been encoded either directly in the form of graphs with an editor or in the form of syntactical matrices then semi-automatically converted into graphs according to E. Roche (1993). These studies led to develop new conversion algorithms in the case of matrix systems where linguistic information is encoded in several matrices.

For the second issue, a prototype on-line library of local grammars have been designed and implemented. The objective is to centralize and distribute local grammars constructed within the RELEX network of laboratories. We developed a set of tools allowing users to both store new graphs and search for graphs according to different criteria. The implementation of a grammar search engine led to an investigation into a new field of information retrieval: searching of linguistic information into sets of local grammars.

Keywords: conversion of syntactical matrices into graphs, expression of measurement, geographical proper noun, graph, lexicon-grammar, linguistic information retrieval, local grammar, locative prepositional phrase, management of local grammars, recursive transition network, syntactical analysis, text parsing

Sommaire

Liste des figures

Liste des tables

Chapitre 1 : Introduction

Ces dernières années, le Traitement Automatique des Langues (TAL) est devenu une discipline incontournable. En effet, l'explosion du nombre de documents disponibles (notamment sur Internet) et de services proposés a rendu nécessaire l'implantation d'outils manipulant des données écrites et orales. Ces outils servent notamment à améliorer l'accès à l'information, comme la recherche de documents, la traduction ou le résumé de textes, etc. (J-M. Pierrel, 2000). La grande majorité des approches proposées utilisent des modèles statistiques qui ont rapidement donné des résultats très prometteurs et à faible coût[1]. Cependant, de plus en plus de chercheurs estiment que de telles approches pourraient rapidement atteindre leurs limites car elles ne prennent pas ou peu en compte le contenu linguistique des données traitées[2] (A. Abeillé et P. Blache, 2000 ; M. Gross et J. Senellart, 1998). Ils marquent l'importance de la linguistique dans ce domaine, surtout du lexique pour l'analyse syntaxique (préalable incontournable à toute analyse sémantique). Ils préconisent la construction de larges bases de descriptions linguistiques, composées de lexiques et de grammaires, quitte à terme à être utilisées dans des modèles statistiques. Ainsi, nous avons assisté au développement de grandes bases de données linguistiques, pouvant s'insérer à différents niveaux de l'analyse automatique (morphologique, syntaxique, sémantique et discursif) : des dictionnaires électroniques ont été construits ; des phénomènes linguistiques très précis sont décrits sous la forme de grammaires locales, les constructions de prédicats sont formalisées dans des tables syntaxiques (M. Gross 1975) ou autres formalismes plus complexes (par exemple, A. Abeillé, 1991) ; des réseaux sémantiques voient le jour, etc. De telles bases de données sont le fruit de travaux collectifs concertés dans un cadre formel précis et cohérent où chaque linguiste apporte sa pierre à l'édifice.

[1] Pour plus de détails sur les techniques statistiques, voir J. Allen (1995), S. Abney (1996), E. Charniak (1997), etc.

[2] Les modèles statistiques sont en fait des modèles linguistiques primaires (et partiels) (S. Abney, 1996) car ils utilisent implicitement des phénomènes réels de la langue (Voorhees, 1999) : ils réalisent des mesures quantitatives de combinaisons de mots et même de catégories linguistiques (grammaticales, syntaxiques ou sémantiques) dans des textes.

Cette démarche a, cependant, un gros inconvénient : elle nécessite un investissement lourd qui s'inscrit sur le long terme. Décrire ou répertorier précisément tous les phénomènes de la langue est une tâche qui requiert du temps : par exemple, la construction des tables syntaxiques des prédicats du français a commencé au début des années 1970 par M. Gross (1975) et son équipe (C. Leclère et al., 1991) et n'est pas achevée. Par ailleurs, ces approches purement linguistiques n'ont pas d'applications impressionnantes à proposer à court terme. En effet, bien qu'il existe de grandes bibliothèques de descriptions formelles et lexicales, le plus souvent, elles ne sont pas exploitables directement et nécessitent de longues opérations de reformatage et de conversion en données applicables. De plus, les méthodes linguistiques, tout en permettant de repérer des phénomènes très précis, génèrent beaucoup de bruit du fait de l'ambiguïté naturelle de la langue qui doit être prise en compte. L'élimination des ambiguïtés (grammaticales, syntaxiques et sémantique) par des méthodes exactes[3] n'en est qu'à ses balbutiements.

De gros efforts sont faits pour estomper les défauts des méthodes linguistiques. D'abord, des méthodes formelles et systématiques de description basées sur l'observation taxonomique des faits linguistiques ont été élaborées : par exemple, la méthodologie du lexique-grammaire de M. Gross (1975). On assiste, par ailleurs, à la naissance d'outils informatiques facilitant le repérage de phénomènes linguistiques complexes dans les textes : par exemple, Intex (M. Silberztein, 1993), Unitex (S. Paumier, 2003), etc. Ensuite, de vastes corpus annotés ou non sont constitués afin d'offrir un champ d'investigation et d'évaluation plus grand[4] (J. Veronis 2000): par exemple, le Brown Corpus, LOB corpus, British National Corpus (Garside et al. 1987), Penn Treebank (M. P. Marcus et al, 1993) pour l'anglais ; Frantext (P. Bernard et al., 2002), Abeillé et al. (2001) pour le français. Enfin, des études sur la normalisation et la diffusion des ressources construites (corpus, dictionnaires, etc.) sont en cours (Romary, 2000). Ces différents efforts doivent aboutir à une meilleure collaboration et à des échanges de données. C'est un secteur clé en devenir de l'informatique linguistique. Pour construire leurs propres « briques », les linguistes ont souvent besoin de celles des autres.

Notre travail s'inscrit dans une approche linguistique du TAL et plus particulièrement de l'analyse automatique des textes. Nous nous plaçons dans le cadre méthodologique du lexique-grammaire (M. Gross, 1975). Nous souhaitons apporter notre contribution sur deux points :

- La description de processus complets de construction de composants linguistiques directement applicables à des textes.

- La conception d'outils informatiques de gestion et de diffusion de larges bibliothèques de données linguistiques.

Nous nous consacrons essentiellement à un type précis de données : les grammaires locales (GL). Elles servent à localiser des phénomènes locaux de manière très précise dans les textes, comme les dates (D. Maurel, 1990), les déterminants numéraux (M. Silberztein 1993, A. Chrobot, 2000), les incises (C. Fairon, 2000). Ces grammaires sont des graphes lexicalisés (M. Silberztein, 1993 ; M. Gross, 1997) qui font appel à des dictionnaires de mots simples (B. Courtois et M. Silberztein, 1990) et de mots composés (B. Courtois et al., 1997). Elles sont

[3] Elles sont à opposer aux méthodes approximatives (ou statistiques) qui génèrent des erreurs.

[4] La constitution de corpus est aussi d'un grand recours pour les approches statistiques pour la phase d'apprentissage. Plus les corpus sont grands, variés et de qualité, plus la précision des statistiques a des chances d'approcher la réalité linguistique.

équivalentes à des réseaux récursifs de transitions [RTN][5] (W. Woods, 1970). Elles ont le grand avantage de pouvoir être appliquées directement et efficacement à des textes (M. Silberztein, 1993 ; S. Paumier, 2000).

D'abord, nous montrons, dans leur totalité, deux processus de construction de GL représentant respectivement des expressions de mesure et des adverbes locatifs. Nous verrons que ces exemples particuliers couvrent de très nombreux champs de difficultés : notre travail a donc un intérêt méthodologique certain. La construction de GL est toujours précédée d'une analyse linguistique détaillée du phénomène que l'on souhaite représenter. Il existe en gros deux méthodes :

- Construction directe et artisanale à l'aide d'un éditeur de graphes ;

- Construction indirecte à l'aide de tables syntaxiques codées à la main et d'une méthode semi-automatique de conversion des tables en GL.

Dans la plupart des cas, lorsque l'on utilise la méthode indirecte, le phénomène étudié est codé dans une simple table syntaxique, mais nous verrons qu'il est parfois préférable de représenter les contraintes dans des systèmes de tables relationnelles.

Enfin, nous abordons le thème de la gestion de larges bibliothèques de grammaires locales. Le nombre de GL augmente vertigineusement. Par ailleurs, le formalisme très modulaire des GL rend possible leur réutilisation dans d'autres GL[6], ce qui facilite le travail en équipe. Nous proposons d'implanter un système centralisé de stockage et de diffusion adéquat. Nous verrons que cela n'est pas évident car les grammaires sont des objets relativement complexes. Les problèmes algorithmiques sont plus difficiles qu'ils n'en ont l'air à première vue. Les algorithmes de stockage (ajout et suppression) de grammaires dans une bibliothèque manipulent la théorie des graphes. Les outils de recherche d'information n'utilisent pas des requêtes standards pour les bases de données car on veut pouvoir rechercher des informations précises sur le contenu linguistique des GL de la bibliothèque. Alors que la communauté « TAL » commence juste à s'intéresser à la construction d'outils de gestion de ressources linguistiques, les études et projets sont limités à quelques types de ressources : logiciels, corpus ou dictionnaires. L'intérêt de notre travail apparaît donc clairement.

[5] Le plus souvent, on peut construire des automates finis à partir de ces RTN.
[6] En pratique, les GL sont des graphes qui ont la propriété de pouvoir appeler des sous-graphes indépendants (formalisme des RTN).

Chapitre 2 : Lexique-grammaire et grammaires locales

2.1 Notations

Les notations ci-dessous serviront tout au long de ce mémoire :

P	Phrase
V	Verbe
Vsup	Verbe support
W	Ensemble des compléments essentiels d'un prédicat
*N*0, *N*1, *N*2	Groupes nominaux arguments d'un prédicat
GN	Groupe Nominal libre
GNloc	Groupe nominal locatif
N	Nom
Nc	Nom classifieur de lieu
Npr	Nom propre
Adj	Adjectif
Modif	Modifieur
Det	Déterminant
Dnum	Déterminant numérique
Prep	Préposition
Loc	Préposition locative
E	Mot vide

2.2 Introduction

Notre travail s'inscrit dans le cadre théorique du lexique-grammaire. Le lexique-grammaire est d'abord une approche formelle, transformationnelle et empirique de la linguistique qui met en avant le caractère fondamental du lexique. L'objectif est de recenser exhaustivement et systématiquement l'ensemble des comportements syntaxiques des phrases simples. Elle se démarque notamment de la très populaire grammaire générative de N. Chomsky (1957) dont le but est de trouver un système abstrait et cohérent décrivant une grammaire universelle.

Grâce à ses caractéristiques, le lexique-grammaire a montré son intérêt certain pour quelques domaines de l'analyse automatique des textes comme l'analyse lexicale ou l'analyse syntaxique du fait de l'accumulation systématique de composants linguistiques. La démarche adoptée est à l'opposé des approches statistiques majoritairement utilisées dans ce domaine. Nous insisterons notamment sur un type particulier de composant : les grammaires locales qui décrivent des phénomènes linguistiques locaux de manière compacte et précise.
Dans ce chapitre, nous détaillons les points fondamentaux du lexique-grammaire : la méthodologie de M. Gross (1975) qui prône l'accumulation systématique des faits linguistiques ; les différents composants linguistiques accumulés depuis une trentaine d'années (dictionnaires morphologiques et syntaxiques, grammaires locales). Puis, nous donnons quelques généralités sur l'analyse automatique des langues et nous répertorions les applications du lexique-grammaire dans ce domaine. Enfin, nous faisons un état de l'art sur les grammaires locales construites au sein de la communauté travaillant dans le cadre du lexique-grammaire.

2.3 Le lexique-grammaire

Le lexique-grammaire est une méthodologie dont l'ouvrage fondateur est « Méthodes en syntaxe » de Maurice Gross (1975). Elle est en grande partie tirée des travaux sur les grammaires transformationnelles de Z.S. Harris (1951,1968) qui a introduit une approche mathématique de la linguistique avec des définitions rigoureuses et minimales. M. Gross a montré l'importance du lexique (souvent déprécié au profit de la grammaire) en montrant par l'accumulation systématique de faits linguistiques que les règles de grammaire (même les plus simples) ne sont pas aussi régulières que l'on a tendance à le croire.

2.3.1 L'objet d'étude : les phrases simples

Les travaux du lexique-grammaire ont pour objet d'étude les phrases simples (ou élémentaires) composées d'un prédicat (verbe, adjectif et, par extension, nom, adverbe) et d'arguments. Le prédicat est l'élément central de la phrase. Son premier argument est le sujet de la phrase. Tout prédicat a un sujet qui est :

- soit un syntagme nominal noté *N*0 :

(Max + La soliste + Il) chante[7]

- soit une complétive notée *que P* (ou une infinitive souvent réduction de la complétive) :

Que Max vienne dérange Léa

- soit un sujet impersonnel :

Il pleut

Les autres arguments sont les compléments essentiels du prédicat : en général, des compléments d'objets. Ces compléments d'objets répondent à la question en *(à + de + E) (quoi + qui)* :

Max donne une pomme à Marie

[7] Le symbole + est le symbole du OU logique.

Que donne Max à Marie ? une pomme
A qui Max donne une pomme ? à Marie

On note *N*1, *N*2 les compléments nominaux essentiels d'un prédicats (respectivement, objets directs et objets indirects). Les compléments essentiels peuvent être des complétives ou des infinitives qui répondent aussi à ces questions :

Luc dit à Paul que Max pleure
Que dit Luc à Paul ? que Max pleure

Marie veut se lever de bonne heure
Que veut Marie ? se lever de bonne heure

Ces règles ne sont pas toujours valides, il existe toujours des exceptions, par exemple avec le verbe *aller* :

Max va courir dans les bois
**Que va Max ? courir dans les bois*[8]

Cet emploi du verbe *aller* répond à la question en *où*.

Où va Max ? courir dans les bois

La notion de phrase élémentaire est une notion plus empirique que théorique. Au fur et à mesure de l'examen des prédicats, on s'aperçoit que cette notion ne peut-être totalement fixée. A. Guillet et C. Leclère (1992) considèrent que certains compléments locatifs sont compléments de prédicats (verbaux, dans leur cas). En effet, la construction locative suivante est équivalente à une phrase simple :

Luc charge les caisses dans le camion
= Luc charge le camion de caisses

Par analogie, ils considèrent certaines constructions locatives de même surface comme des phrases simples, bien qu'elles n'aient pas de phrases simples traditionnelles équivalentes :

Luc plonge les légumes dans l'évier
*= *Luc plonge l'évier de légumes*

Ces dernières constructions sont considérées comme élémentaires car le complément locatif dépend du verbe et non de la phrase comme pour les compléments circonstanciels. On illustre cela avec la phrase *Marie danse dans le salon* :

(Que Marie danse+ la danse que fait Marie) se déroule dans le salon
**Que Max plonge les légumes se déroule dans l'évier*

Comme l'indique B. Lamiroy (1999), les travaux les plus poussés ont été réalisés sur les verbes car les constructions simples à prédicat verbal sont les plus naturelles. Mais, de nombreuses études ont été menées sur d'autres parties du discours. Lorsque les prédicats sont

[8] Le symbole * devant une séquence indique que cette dernière est interdite.

des noms, ils sont toujours accompagnés d'un verbe support sémantiquement « neutre » comme *faire* (J. Giry-Schneider 1978, 1987) ou *avoir* (J. Labelle, 1974), etc.[9] :

Max fait une injure à Léa
Max fait un livre sur Paul
Luc a une correspondance avec Léa

Les adjectifs peuvent être des prédicats comme *soucieux, joli.* Ils entrent dans des constructions en *être* (A. Meunier 1981 ; E. Laporte 2002) :

Max est soucieux de ce que Léa réussisse son examen
Marie est jolie

Les adverbes traditionnels sont quant à eux un peu à part dans les phrases car ils sont généralement syntaxiquement indépendants du reste. Cependant, ils s'interprètent aussi à l'aide d'une phrase élémentaire à verbe support d'adverbe :

Hier, Max a tué un éléphant
= Max a tué un éléphant ; cela s'est produit hier

Les adverbes au sens de M. Gross (1986) rentrent également dans des constructions à verbes supports comme nous l'avons brièvement évoqué ci-dessus (L. Danlos 1980 ; M. Gross 1986, 1996) :

La détonation s'est produite sur le coup de minuit
Ces recherches sont à la pointe du progrès

Les dernières expressions ci-dessus sont figées. Elles ne présentent aucune différence en surface avec des expressions dites libres, si ce n'est que certains de leurs éléments sont contraints au niveau lexical, syntaxique ou sémantique. Les adverbes ne sont pas les seuls à rentrer dans des constructions figées ou semi-figées (M. Gross, 1984 ; J. Giry-Schneider, 1978) :

Luc fait face à un problème
Luc a les yeux en compote
Marie prend la poudre d'escampette
Il fait chaud

2.3.2 La phrase simple comme unité élémentaire de sens

Inspiré par Z.S. Harris, M. Gross considère la phrase simple comme l'unité élémentaire de sens, une phrase simple étant formée d'un prédicat et de ses arguments obligatoires. Cette affirmation paraît à première vue exagérée mais elle se vérifie dans la plupart des cas.
Tout d'abord, un type de phrase satisfait parfaitement cette affirmation : ce sont les phrases figées. Par exemple, l'emploi figuré de *Max prend la porte* ne peut être interprété sémantiquement qu'en prenant la phrase dans son ensemble.
Pour les phrases libres, cela semble aussi se vérifier. Par exemple, prenons le verbe *voler* qui a deux emplois : l'un équivalent au verbe *to steal* en anglais, l'autre équivalent à *to fly*. Le premier emploi entre dans une structure simple sans complément (*L'oiseau vole*). Le

[9] Cf. D. de Negroni-Peyre (1978), A. Meunier (1981) , R. Vivès (1983), G. Gross (1989).

deuxième emploi entre dans une construction simple avec un complément d'objet direct et un complément d'objet indirect : *Max vole une montre à Léa*. Ainsi, le sens d'une occurrence du verbe *voler* ne peut être calculé que si l'on examine la phrase élémentaire dans laquelle il rentre. Il en est de même avec les noms prédicatifs : le nom *fête*, par exemple, a plusieurs emplois (ou sens) qui se distinguent par la phrase élémentaire dans laquelle ils rentrent.

Marie fait la fête
Marie fait sa fête à Luc
etc.

A un niveau plus sémantique, si l'on prend le verbe *couvrir* par exemple qui comporte deux compléments essentiels, le complément d'objet indirect peut être omis. Cependant, il est toujours sous-entendu que l'on *couvre quelque chose de quelque chose* comme dans :

Max couvre le toit de feuilles

Le postulat de M. Gross semble à première vue ne pas se vérifier pour les mots non-prédicatifs. Mais, en les insérant dans des phrases, on observe que leur interprétation dépend du contexte.

Luc aime (le cheval + manger de la viande de cheval + faire du cheval)

Ces mots constituent la majorité des mots et sont souvent des noms (*voiture, cheval, table, main, Paul)*.

La voiture percute le cheval

Certains adjectifs sont également non-prédicatifs car ils ne rentrent pas dans des constructions en *être* :

Max a gravi la face nord de l'Everest
**La face de l'Everest est nord*

2.3.3 Transformations et génération de phrases complexes

Les phrases simples sont sujettes à des transformations qui consistent en une succession d'opérations élémentaires (ex. effacement, substitution, déplacement, etc.). Par exemple, la pronominalisation du complément d'objet de *Max mange une pomme* consiste à substituer *une pomme* par *la* (*Max mange la*) puis à déplacer *la* pour former la phrase *Max la mange*. Ces opérations étant réversibles, les transformations sont non-orientées. Les deux phrases précédentes sont dites équivalentes et sont mises en relation par le signe '=' :

Max mange une pomme
= Max la mange

Dans la grande majorité des cas, ces transformations n'introduisent pas de changements sémantiques significatifs même s'il peut exister quelques nuances, comme la négation qui, généralement, donne un sens opposé à la phrase d'origine :

Max mange une pomme
= Max ne mange pas (une + de) pomme

On dira que ces deux phrases sont à peu près équivalentes.

Il existe deux grands types de transformations : les transformations unaires et les transformations binaires. Les transformations unaires les plus connues sont la passivation, la pronominalisation, l'extraposition (M. Gross, 1990) :

Max a acheté beaucoup d'engrais
= Il a acheté beaucoup d'engrais (pronominalisation du sujet)
= Il en a acheté beaucoup (pronominalisation *en* du complément d'inclusion)

Max a acheté beaucoup d'engrais
= Beaucoup d'engrais a été acheté par Max (passif)
= Il a été acheté beaucoup d'engrais par Max (extraposition)
= Il en a été acheté beaucoup par Max (pronominalisation *en* du complément d'inclusion)

Cet ensemble de phrases constitue une classe d'équivalence. Les classes d'équivalences peuvent être étendues à l'aide de transformations entre différentes parties du discours. Par exemple, il existe une relation d'équivalence entre la classe du verbe *injurier* et celle du nom *injure* par la transformation de nominalisation (J. Giry-Schneider, 1978) :

Max injurie Léa
= Max fait des injures à Léa

Meunier (1981) a étudié les transformations d'adjectivation entre les noms et les adjectifs :

Max est soucieux de ce que Léa réussisse son examen
= Max a le souci de ce que Léa réussisse son examen

Les transformations binaires opèrent sur deux phrases simples. Ce sont par exemple :

- la coordination :

  Max travaille et Luc joue (*= Max travaille ; Luc joue*)

- la subordination circonstancielle :

  Max travaille pendant que Luc joue

- la relativation

Luc aime la femme que Paul a dénoncée à la police

En fait, les phrases simples servent à générer des phrases complexes au moyen des transformations. Elles sont combinées entre elles à l'aide des transformations binaires. La réduction de constructions à prédicats nominaux ou adjectivaux permet de compacter les informations d'une phrase simple en structures nominales et de les insérer dans des phrases :

Le livre de Max sur Paul est un gros travail

= *Max fait un livre sur Paul* ; *cela est un gros travail*

Cet homme gentil va se faire exploiter
= Cet homme est gentil ; il va se faire exploiter

2.3.4 Une démarche expérimentale

L'un des buts de la linguistique est de classer les objets linguistiques, notamment les prédicats. Dans la plupart des études, les prédicats sont classés selon des concepts sémantiques très difficiles à définir à l'aide de critères formels. M. Gross a fait le choix de regrouper les verbes selon leur structure définitionnelle (cf. M. Gross, 1975), c'est-à-dire selon leur forme de surface :

[Verbes : *N*0 *V W*]

*N*0 *V* =: *Paul pleure*
*N*0 *V N*1 =: *Marie aime le golf*
*N*0 *V N*1 *à N*2 =: *Luc offre des fleurs à Léa*
*N*0 *V N*1 *Loc N*2 =: *Max plonge sa main dans une casserole d'eau fraîche*
…
*Que P V N*1 =: *Que Luc parte intrigue Marie*
*N*0 *V que P à N*1 =: *Max répète qu'il n'est pas content à Luc*

[Noms : *N*0 *Vsup Prep Det N W*]

*N*0 *avoir Det N* =: *Max a du pouvoir*
…
*N*0 *faire Det N* =: *Luc fait (du stop + un aller simple)*
*N*0 *faire Det N Prep*1 *N*1 =: *Luc fait des investigations sur Léa*
…
*N*0 *être Prep Det N* =: *Paul est à son avantage*
*N*0 *être Prep N Prep*1 *N*1 =: *Marie est en avance sur Max*

Dans sa conclusion, M. Gross (1975) estime que certaines classes sont sémantiquement homogènes comme

- la table 2 dont les verbes suggèrent une idée de « mouvement » : *arriver*, *courir*, *couler*, etc. (*N*0 *V V-inf W*)
- la table 9 qui contient les verbes de communication : *annoncer*, *confier*, *dire*, *rapporter*, *vociférer*, etc. (*N*0 *V que P à N*2)
- la table 12 dont les verbes dénotent une idée de « jugement de valeur » : *adorer*, *critiquer*, *réprimander*, *soutenir*, etc. (*N*0 *V N*1 *du fait que P*)

L'originalité des travaux de M. Gross et de son équipe réside dans sa démarche formelle, expérimentale et systématique. En effet, il considère qu'à l'instar des autres sciences, c'est à partir d'expériences que l'on conçoit les théories. La méthodologie du lexique-grammaire part des faits linguistiques observables pour trouver un modèle linguistique tout en s'appuyant sur les transformations et opérations formelles harrissiennes. Par exemple, chaque prédicat est placé dans une construction simple et est systématiquement soumis à des expériences. Ces expériences consistent à appliquer à la phrase différentes transformations et à décider l'acceptabilité de la phrase obtenue. Le fait d'utiliser des outils formels rend ces expériences reproductibles. Les travaux effectués ont notamment permis de découvrir qu'il n'existe

pratiquement pas de verbes ayant exactement le même comportement syntaxique. Les prédicats de même construction définitionnelle possèdent des propriétés syntaxiques communes mais présentent aussi des différences de comportement vis-à-vis des transformations. Par exemple, les contraintes distributionnelles des sujets peuvent diverger :

(Cet événement + Luc) révèle à Paul que sa femme le trompe
*(*Cet événement + Luc) écrit à Paul que sa femme le trompe*

Certains verbes prenant un complément d'objet direct n'acceptent pas la transformation de passivation :

*Le pont coûte beaucoup d'argent = *Beaucoup d'argent est coûté par le pont*
Le pont attire les touristes = Les touristes sont attirés par le pont

Certains verbes acceptent des dérivations morphologiques ; d'autres pas :

L'eau tourbillonne = L'eau fait des tourbillons
*L'eau bouge = *L'eau fait des bouges*

La distribution des déterminants est au centre des études sur les noms prédicatifs. Elle dépend de chaque substantif :

*Luc fait (*un + du + *des) stop*
*Luc fait (*une + *de la + des) vague(s)*
*Luc fait (un + *du + des + son) testament(s)*

Cette méthodologie a permis de mettre en avant le rôle fondamental du lexique. Cette approche systématique est un gros investissement et nécessite du temps et du travail d'équipe. Grâce à son caractère formel et expérimental, une telle méthode est reproductible dans d'autres langues que le français. Ainsi, il existe un large réseau de laboratoires utilisant le lexique-grammaire. Cette communauté se réunit annuellement lors du colloque « grammaires et lexiques comparées » depuis plus de vingt ans.

2.3.5 Des composants linguistiques de haute précision

La constitution de composants linguistiques (lexiques et grammaires) est dans la nature même du lexique-grammaire. L'étude systématique des prédicats a conduit M. Gross et son équipe à représenter le comportement syntaxique des prédicats dans des tables syntaxiques. Chaque table contient en gros les prédicats de même structure de surface. Chaque ligne correspond à une entrée lexicale (ou un prédicat). Chaque colonne correspond à une propriété. Toutes les propriétés ne sont pas représentées car certaines sont communes à toutes les entrées de la table. A l'intersection d'une ligne (une entrée) et d'une colonne (propriété), il y a un signe + si l'entrée lexicale accepte cette propriété ; un signe – si elle ne l'accepte pas ; une information lexicale si besoin est. Ces tables sont appelées communément tables de lexique-grammaire (parfois, tables ou dictionnaires syntaxiques). Les chercheurs du lexique-grammaire ont ainsi codé 12 000 emplois de verbes (M. Gross, 1975 ; J.P. Boons, A. Guillet et C. Leclère, 1976 ; A. Guillet et C. Leclère, 1992), 10 000 emplois de noms prédicatifs (J. Giry-Schneider, 1978, 1987 ; J. Labelle, 1974 ; A. Meunier, 1981 ; R. Vives, 1983 ; D. de Negroni, 1978 ; G. Gross 1989). Les tables des adjectifs sont en cours de construction. Il existe également des tables de phrases figées (M. Gross, 1984) comprenant une vingtaine de

milliers d'entrées. Ces tables servent de base à un travail de comparaison entre différentes variantes du français (BFQS = Belge Français Québécois Suisse).

N0 = Nhum	N0 = Nnr	N0 = V-n	N1 V	Aux = avoir	<ENT>	N1 = V-n	N0 V N1 de combien	N0 V N1 dans N2	N0 V Nhum sur ce point	N0 est Vpp_W	N1 = Nconc	N1 = Nabst	<OPT>V-n (N0)	<OPT>V-n (N1)	<OPT>Exemple
+	+	-	-	+	abandonner	-	-	-	-	-	-	-	-	-	La chance§abandonne§Max
-	+	-	-	+	abasourdir	-	-	-	-	+	-	-	-	-	Le bruit a§abasourdi§Max
+	-	-	-	+	abattre	-	-	-	-	-	-	-	-	-	La police a§abattu§le truand*
+	-	-	-	+	abattre	-	-	-	-	+	-	-	-	-	Le peuple§abattra§le tyran*
+	-	-	-	+	aborder	-	-	-	-	-	-	-	-	-	Max a§abordé§une dame dans la rue
+	-	-	-	+	accolader	-	-	-	-	-	-	-	-	-	Le ministre a§accoladé§le héros
+	-	-	-	+	accompagner	-	-	-	-	+	-	-	-	-	Max§accompagne§Léa
+	-	-	-	+	accoster	-	-	-	-	-	-	-	-	-	Max a§accosté§une dame dans la rue
+	-	-	-	+	accrocher	-	-	-	+	-	-	-	-	-	Jean a§accroché§une nana dans la rue*
+	-	-	-	+	accrocher	-	-	-	-	-	-	-	-	-	Les guerilleros ont§accroché§les soldats dans le défilé*
+	-	-	-	+	acheter	-	-	-	-	+	-	+	-	-	Max a§acheté§un député
+	-	+	-	+	administrer	+	-	-	-	+	-	+	administrateur	administré	Ce fonctionnaire§administre§un grand nombre d'employés
+	-	-	-	+	adopter	-	-	-	-	+	+	+	-	-	Paul a§adopté§un petit Hindou
+	-	-	-	+	agrafer	-	-	-	-	-	-	-	-	-	Ce raseur a§agrafé§Max dans la rue
+	+	-	-	+	agresser	-	-	-	+	-	-	+	-	-	Max a§agressé§une passante
+	-	-	-	+	aimer	-	-	-	-	+	-	-	-	-	Max§aime§Ida
+	-	-	-	+	alarmer	-	-	+	+	-	-	-	-	-	Jo a§alarmé§les pompiers
+	+	-	-	+	aliter	-	-	-	-	+	-	-	-	-	Max a§alité§Luc|*Loc N=lit|
+	-	-	-	+	allumer	-	-	-	-	-	-	-	-	-	Max a§allumé§Luc à la carabine
+	-	-	-	+	alpaguer	-	-	-	-	-	-	-	-	-	La police a§alpagué§Max
+	-	-	-	+	alphabétiser	-	-	-	-	+	-	-	-	-	Max§alphabétise§les immigrés
+	-	-	-	+	amorcer	-	-	+	+	-	-	-	-	-	Max a§amorcé§les truites la veille
+	-	+	-	+	analyser	-	-	+	+	+	-	+	+	-	Sigmund§analyse§Flora
+	-	-	-	+	anathématiser	-	-	-	-	+	+	+	-	-	Max§anathématise§les femmes
+	-	+	-	+	anesthésier	-	-	-	-	+	-	-	anesthésiste	-	Le chirurgien a§anesthésié§Max
+	-	-	-	+	apostropher	-	-	-	+	-	-	-	-	-	Max a§apostrophé§Luc

Table 1 : extrait de la table 32H (J.P. Boons, A. Guillet, C. Leclère, 1976b)

Certaines expressions figées ou semi-figées telles que les dates sont plus adéquatement décrites sous la forme d'automates finis (plus communément appelés graphes dans la communauté du lexique-grammaire). Cette représentation est très naturelle et sa lecture immédiate (si les graphes sont bien conçus). Nous donnons un exemple de grammaire anglaise de dates ci-dessous. Elle reconnaît des expressions telles que *five in the afternoon*, *5 p.m.*, *half past one*. Nous reviendrons sur ce type de données car c'est le sujet central de ce mémoire.

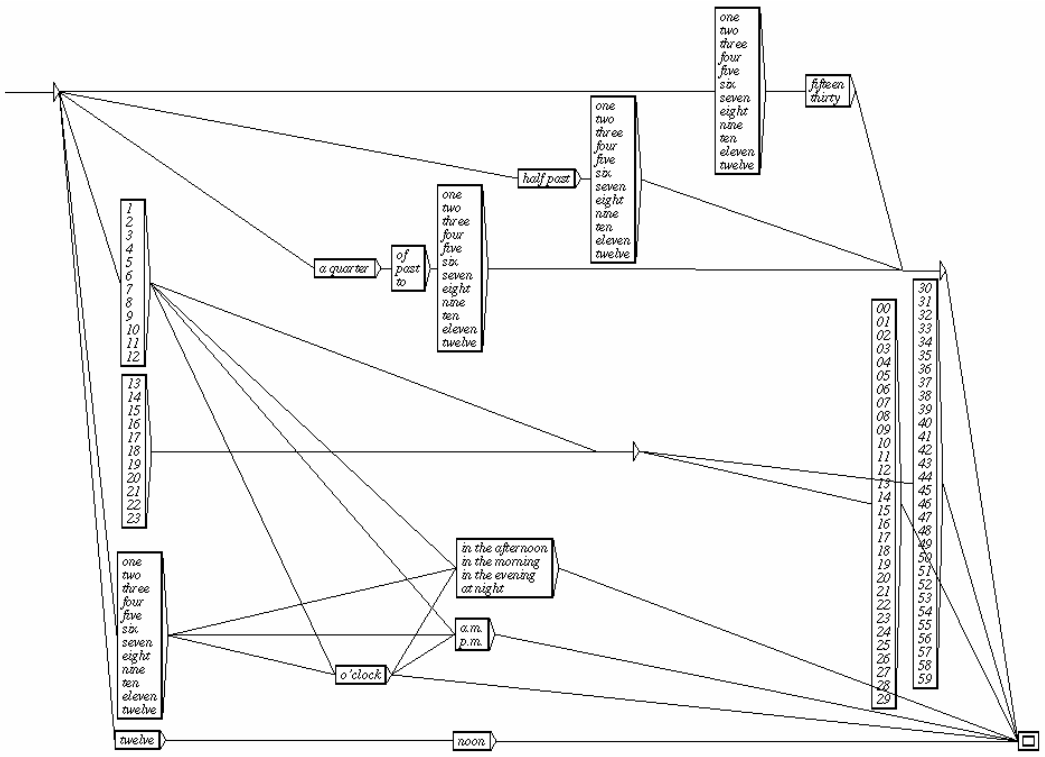

Figure 1 : exemple de grammaire locale de dates en anglais

Dans un souci d'exhaustivité, les chercheurs du lexique-grammaire ont également construits des dictionnaires électroniques de formes fléchies de mots simples (DELAF) et de mots composés[10] (DELACF). Ces formes fléchies sont automatiquement générées à partir de leur forme canonique et d'une classe flexionnelle associée. A chaque entrée lexicale, est associée un lemme, un code grammatical, des informations flexionnelles (genre, nombre), des traits syntactico-sémantiques et, dans les versions les plus récentes, le nom de la table de lexique-grammaire à laquelle elle se rattache. Nous donnons ci-dessous un exemple d'une entrée ambiguë du DELAF *avions* et d'une entrée du DELACF *pommes de terre* :

avions,avoir.V:I1p
(verbe *avoir* conjugué à l'imparfait à la troisième personne du pluriel)
avions,avion.N:mp
(nom *avion* au masculin pluriel)
pommes de terre, pomme de terre.N+NDN:fp
(nom composé *pomme de terre* au féminin pluriel)

Nous notons qu'il existe aussi des dictionnaires phonétiques DELAP (E. Laporte, 1988).

[10] Les mots composés sont des séquences de mots simples qui forment une unité linguistique (cf. section suivante).

2.4 Lexique-grammaire et analyse automatique de textes

Les composants linguistiques accumulés dans le cadre du lexique-grammaire sont sous la forme de représentations formelles simples : listes, tables, automates finis et réseaux récursifs de transitions. Leur intégration dans des applications du domaine du Traitement Automatique des Langues (TAL) est donc naturelle. Dans cette section, nous réalisons un bref panorama du domaine de l'analyse automatique de textes. Puis, nous montrons ce que peuvent apporter les composants linguistiques accumulés dans le cadre du lexique-grammaire.

2.4.1 L'analyse automatique de textes

Le Traitement Automatique des Langues (TAL) traite deux types d'objets linguistiques : des flux textuels et des flux de paroles. Ces objets servent d'entrées/sorties à des systèmes d'analyse et à des systèmes de génération. Un processus d'analyse reçoit en entrée un objet linguistique et son but est de donner en sortie une représentation formelle de cet objet : catégories grammaticales des mots, structure syntaxique des phrases, représentation du sens, etc. A l'opposé, un processus de génération prend en entrée une représentation formelle abstraite et construit à partir de celui-ci un objet linguistique. Les applications du TAL telles que la traduction automatique, la reconnaissance vocale et la synthèse vocale combinent ces deux processus. La génération est plus difficile car l'entrée est abstraite et non standardisée donc difficile à cerner (L. Danlos, 1985, 2000). L'analyse part, quant à elle, de données concrètes connues.

Notre travail se situe dans le domaine de l'analyse automatique de textes qui, malgré la relative facilité comparée à la génération, n'est pas une mince affaire. Une application typique de l'analyse est la recherche automatique de documents dans une base de données textuelles. Etant donnée une requête, soit une séquence de mots-clés (le plus souvent des noms), il s'agit de trouver les documents les plus adaptés à la requête en tenant compte de réalités linguistiques et des connaissances du monde. La recherche documentaire [11] utilise des techniques linguistiques comme l'indexation des textes, la lemmatisation des mots du texte (ex. remplacer une forme conjuguée par sa forme à l'infinitif), l'utilisation des synonymes, la levée d'ambiguïté. Cette dernière permet, par exemple, d'éliminer des index les mots grammaticaux initialement ambigus avec des mots pleins (ex : *or*).

L'analyse présente un problème majeur : comment représenter le résultat de manière rigoureuse et formelle ? D'abord, l'analyse et la représentation de son résultat dépendent de l'application que l'on veut en faire : il paraît clair que la finesse d'analyse pour la traduction doit être infiniment plus grande que pour la recherche documentaire. Ensuite, un autre problème de l'analyse est qu'il n'existe pas de représentation standard, même si certains standards existent comme EAGLES (1996), mais cela ne va pas beaucoup plus loin que la syntaxe. L'analyse des mots est la plus aisée car les catégories grammaticales existent et sont admises depuis l'antiquité. Cependant, il existe un problème de taille : l'ambiguïté de la langue. En effet, chaque mot en moyenne peut être interprété de deux façons différentes. L'analyse d'un texte passe obligatoirement par la levée des ces ambiguïtés. Les processus de désambiguïsation permettent d'associer à chaque mot une étiquette. Il existe de nombreuses méthodes qui consistent à examiner le contexte proche du mot traité soit à l'aide de calculs statistiques (K. Church, 1988 ; E. Dermatas et al., 1995) soit à l'aide d'indices ou de règles linguistiques (K. Koskenniemi, 1983 ; M. Silberztein, 1993 ; K. Oflazer, 1996 ; E. Laporte et al., 1999). Certaines applications mélangent les deux approches (E. Brill, 1995). La comparaison des méthodes utilisées est très difficile car les jeux d'étiquettes employés sont très différents, tout dépend de la finesse d'analyse que l'on souhaite avoir (E. Laporte, 2000).

[11] cf. Fluhr (2000) ;T. Strzalkowski et al. (2000) ; E. Voorhees (1999).

L'analyse syntaxique des phrases est particulièrement problématique car il existe un très grand nombre de modèles[12] :

- les grammaires d'unification : grammaire lexicale fonctionnelle (LFG), cf. J. Bresnan et R. Kaplan (1982) ; grammaire syntagmatique guidée par les têtes (HPSG), cf. C. Pollard et I. Sag (1987,1994) ; grammaire d'arbres adjoints (TAG), cf. A. Joshi (1987), etc.
- les grammaires de dépendance (L. Tesnière, 1959 ; I. Mel'cuk, 1988 ; etc.)
- etc.

Cependant, il existe un certain nombre de points où les linguistes sont quasiment unanimement d'accord : on peut regrouper les mots en constituants qui, eux-mêmes, peuvent se regrouper en constituants. On peut donc représenter une phrase par un arbre syntaxique. La nature précise des constituants dans une phrase donnée fait cependant l'objet de controverses entre linguistes : M. Gross (1975, p. 34) a notamment rejeté la notion de groupe verbal (GV). La phrase *Le petit homme mange du pain sec* peut être analysée à l'aide de l'arbre ci-dessous. Les mots sont regroupés sous la forme de syntagmes : *le petit homme* et *du pain sec* sont des groupes nominaux (GN). Le verbe *manger* est le prédicat de la phrase qui a deux arguments : un sujet (*le petit homme*) et un complément d'objet direct (*du pain sec*).

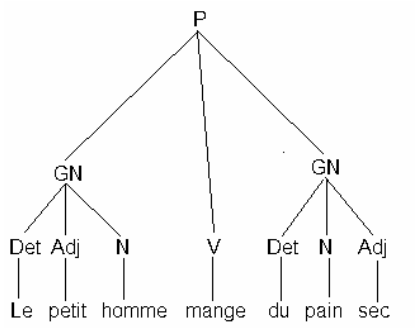

Figure 2 : arbre syntaxique

Nous avons donné dans cet exemple un arbre très simple. Il est possible de faire des analyses avec des arbres indiquant plus d'informations (les noms-têtes des groupes nominaux par exemple). Là aussi, tout dépend de la précision d'analyse que l'on souhaite avoir. L'analyse syntaxique peut jouer un rôle dans la levée d'ambiguïté lorsque deux mots identiques ont la même catégorie grammaticale mais deux sens différents : par exemple, le verbe voler (*to steal* ou *to fly*). L'analyse syntaxique de la phrase dans laquelle il se trouve permet de trouver son sens :

(Paul) vole (un bijou) à (Léa) => to steal
(L'oiseau) vole => to fly

L'analyse syntaxique est primordiale mais n'est pas suffisante. A terme, une analyse sémantique sera nécessaire. L'analyse sémantique des phrases n'en est qu'à ses balbutiements (cf. G. Sabah, 2000). Les ambiguïtés sémantiques lexicales sont parfois traitées au moyen de règles construites automatiquement à partir de corpus (ex : C. Brun, 2000 où le corpus est un dictionnaire). Certains construisent des lexiques généraux sous la forme de classes d'objets (G. Gross, 1994), de réseaux sémantiques (ex. Wordnet par G.A. Miller et al., 1990), etc. Des formalisations sémantiques paraissent cohérentes mais elles sont limitées à des phénomènes bien particuliers (déplacements ou localisation par exemple). Notons que la résolution

[12] Pour plus de détails, voir A. Abeillé (1993) ou A. Abeillé et P. Blache (2000).

d'anaphores jouera certainement un rôle très précieux pour relier les phrases entre elles (R. Mitkov, 2002).

2.4.2 Les solutions du lexique-grammaire

2.4.2.1 La notion de mot

La notion de mot est centrale dans l'analyse de textes. Il est fondamental d'en distinguer deux types : le mot typographique et le mot linguistique. Le mot typographique (ou informatique) est une séquence de lettres de l'alphabet de la langue délimitée par des séparateurs : en français, les espaces, les ponctuations, etc. D'ailleurs, ces mots servent le plus souvent comme base d'indexation d'un texte car cette dernière demande peu d'effort du fait de leur définition purement formelle.

La quasi-totalité de ces mots sont des mots autonomes que l'on trouve dans les dictionnaires : on les appelle des mots simples. Le repérage de tels mots par des procédures de découpage formel n'apporte pas (ou très peu) d'information linguistique. On distingue donc ces mots des mots linguistiques. Les mots que l'on trouve dans les textes sont ce que l'on appelle des formes fléchies de formes canoniques (ou lemmes) : *outils* et *chantaient* proviennent respectivement des lemmes *outil* et *chanter*. Par ailleurs, ils appartiennent à une partie du discours (ou catégorie grammaticale), ont un genre et nombre (et un cas dans certaines langues) pour les noms et les adjectifs, etc.: *outils* est un nom au masculin pluriel et *chantaient* est un verbe à l'imparfait à la troisième personne de pluriel. Un mot linguistique doit contenir explicitement ces informations.

On s'aperçoit rapidement que le découpage des textes en mots simples n'est pas forcément toujours adapté au traitement linguistique. En effet, il est bien connu que la séquence *aujourd'hui* contenant un séparateur de mot « ' » doit être considérée comme une unité car *aujourd* n'est pas un mot autonome (qui peut s'employer seul) du français[13]. Ce type d'unités complexes contenant au moins un séparateur est appelé mot composé. Certains séparateurs sont assez productifs en terme de mots composés comme le tiret « - » : *rez-de-chaussée*. Mais il en existe des milliers contenant d'autres séparateurs comme l'espace en français : *pomme de terre*, *carte de visite*, *fibre optique*. Dans la quasi-totalité des cas, ces mots composés sont formés de mots autonomes du français et ont la même structure de surface que les groupes nominaux libres. Ils se distinguent cependant de ces derniers par leur figement sémantique, syntaxique et lexical. Pour des explications complètes, le lecteur est invité à se référer à G. Gross (1996). Pour beaucoup d'entre eux, le sens global du mot composé n'est pas calculable (ou très difficilement) à partir du sens des mots simples qui la composent : par exemple, un *cordon bleu* (au sens d'excellent cuisinier) n'a rien à voir avec les mots autonomes qui le composent. Leur traduction peut parfois donner une indication sur leur figement : par exemple, *pomme de terre* ne se traduit pas par *apple of ground* ou *ground apple* en anglais mais par *patatoe*. Notons qu'un mot composé comporte aussi des informations linguistiques : *pommes de terre* provient du lemme *pomme de terre* et est un nom au féminin pluriel. Ainsi, le repérage de ce type de mot linguistique est nécessaire pour réaliser une analyse lexicale précise des textes.

2.4.2.2 Analyse lexicale et ambiguïté

Les dictionnaires de mots simples du type DELA (B. Courtois et al., 1990) et de mots composés du type DELAC (B. Courtois et al., 1997) élaborés dans le cadre du lexique-grammaire sont donc d'une utilité remarquable pour ce type d'analyse. En effet, chaque mot du texte peut être associé explicitement à toutes ses informations linguistiques au moyen de

[13] A contrario, la séquence *s'il* doit être découpée en deux mots autonomes (*si il*).

procédures automatiques de consultation des dictionnaires (M. Gross, 1989, M. Silberztein, 1997). Les logiciels Intex (M. Silberztein, 1993) et Unitex (S. Paumier, 2003) ont été conçus à cet effet. Au départ sous la forme de simples listes, les dictionnaires sont compressés sous la forme de transducteurs à états finis minimaux (D. Revuz, 1991). Par exemple, le DELAF français comprenant 900 000 mots a une taille d'environ 1 MO dans sa forme compressée. En plus d'une réduction significative de l'espace mémoire, cette représentation accélère les consultations : en effet, les temps de recherche ne dépendent plus du nombre de mots dans le dictionnaire mais du nombre de lettres du mot à analyser.
Une analyse lexicale aussi précise fait surtout prendre conscience de l'importance de l'ambiguïté de la langue. Même des formes très fréquemment utilisées comme *avions* sont ambiguës : soit nom (*avion*) soit verbe (*avoir*). En moyenne, chaque mot d'un texte du français a deux interprétations. Ainsi, le nombre d'interprétations d'une phrase est exponentielle par rapport au nombre de mots de cette phrase. Si la phrase fait 10 mots, elle aura en moyenne 2^{10} (= 1 024) interprétations possibles. Dans les logiciels Intex et Unitex, le résultat de l'analyse lexicale est représenté au moyen d'un automate afin de compacter la liste des interprétations possibles des phrases. L'analyse de la phrase *les enfants préfèrent les avions à la voiture* est sous la forme de l'automate ci-dessous :

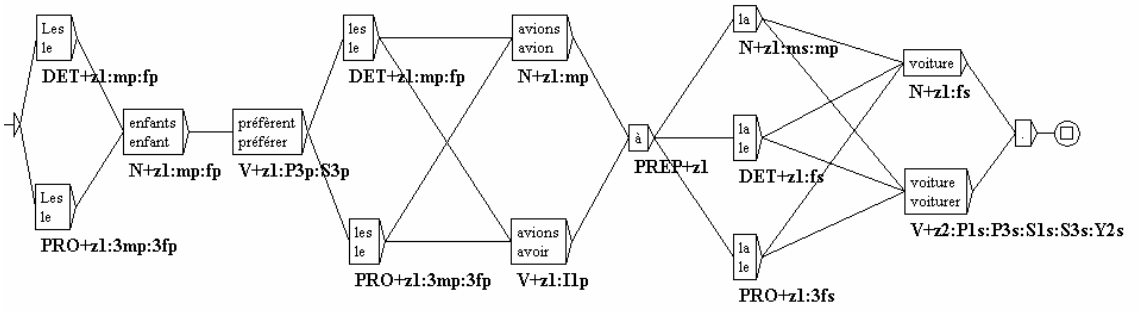

Figure 3 : analyse lexicale de *les enfants préfèrent les avions à la voiture*.

La consultation des dictionnaires de mots composés même si elle apporte une nouvelle analyse, crée de nouvelles ambiguïtés. En effet, la phrase (1) est naturellement ambiguë. L'analyse des mots simples *pomme*, *de* et *terre* du mot composé *pomme de terre* doit être conservée car sinon on ne peut plus analyser (1) comme (1b). Supprimer ces analyses revient à faire un choix d'interprétation (ex : règle de la séquence la plus longue qui est une approximation au même titre qu'une approche statistique [J. Senellart, 1999b]) :

(1) *Le chef recouvre la pomme de terre*
(1a) *(Le chef) recouvre (la pomme de terre)*
(1b) *(Le chef) recouvre (la pomme) de (terre)*

Dans l'interprétation (a), c'est le légume *pomme de terre* que l'on recouvre. L'objet permettant de le recouvrir est omis. Dans (b), c'est le fruit (*pomme*) que l'on recouvre avec de la terre. L'ambiguïté ne peut être levée au niveau syntaxique ; il faut un contexte autour de cette phrase. L'analyse lexicale par consultation des dictionnaires engendre l'automate suivant :

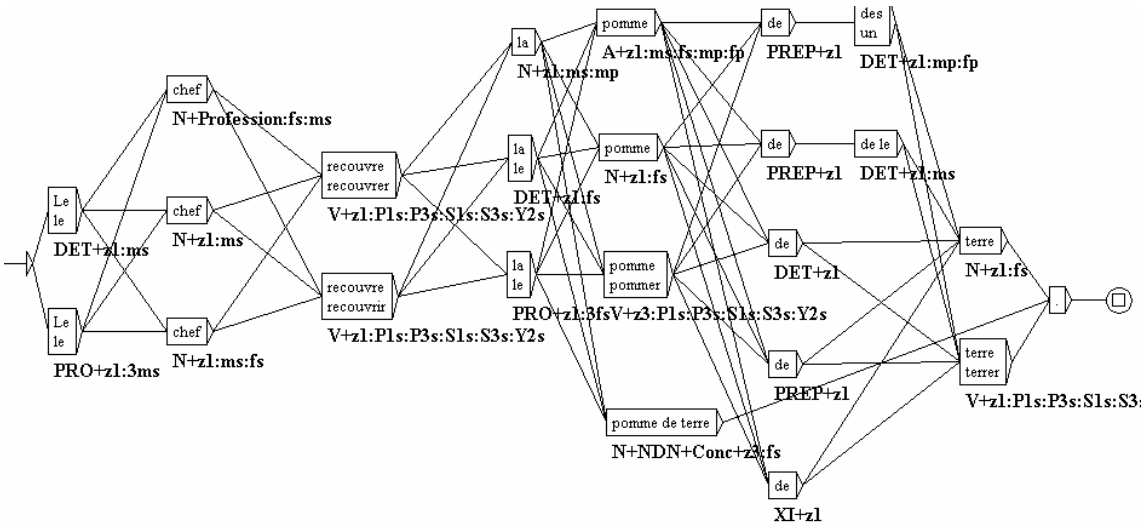

Figure 4 : Analyse lexicale de la phrase *le chef recouvre la pomme de terre*

Comme le montre clairement les automates, l'analyse automatique de textes est rendue ardue du fait de l'ambiguïté colossale de la langue. La levée d'ambiguïté est donc fondamentale. Certains chercheurs proposent de résoudre partiellement ce problème en appliquant des règles concernant des contraintes locales (des séquences limitées à quelques mots). Cette étape intervient juste avant l'analyse syntaxique afin de faciliter cette dernière. Un module de levée d'ambiguïté a été implanté dans Intex. Le logiciel ELAG (E. Laporte et A. Monceaux, 1999) est en train d'être intégré à Unitex. Quelques travaux et de nombreuses règles ont déjà été réalisées (cf. A. Dister, 1999).

2.4.2.3 Entre analyse lexicale et syntaxique

Il est très tentant de considérer certaines séquences de mots comme des mots composés. C'est le cas pour les verbes conjugués à des temps composés tels que *a mangé*. Cependant, ce genre de séquences admettent des insertions de type adverbial :

Max a trop mangé
Max a hier soir mangé comme quatre

Ce phénomène n'est pas observable pour les mots composés : *carte (E + *très) bleue* ; *pomme (E + *cuite) de terre*. Les insertions étant libres, il n'est pas possible de les insérer dans les dictionnaires de mots composés. Le meilleur moyen est d'utiliser des grammaires locales. M. Gross (1999) a notamment réalisé une grammaire reconnaissant les complexes verbaux en anglais du type *is being constructed*, *can eat*, etc. Certaines classes de mots composés sont directement représentées sous la forme d'automates (ou grammaires locales) car cette description est mieux adaptée, plus compacte et plus claire que la description par liste : par exemple, les adverbes composés de dates (D. Maurel, 1990 ; M. Gross, 2002). Certaines expressions figées combinent plusieurs propriétés rendant leur description par dictionnaires électroniques du type DELACF impensable :

- insertion possible
- variation lexicale et syntaxique

Paul a perdu la (boule + tête + raison)
Paul perdit (lentement + après trente ans de mariage) la (boule + tête + raison)

Les grammaires locales sont dès lors requises (J. Senellart, 1999a).

M. Gross et J. Senellart (1998) estiment qu'il pourrait être intéressant de réduire les expressions figées et semi-figées (mots composés, phrases figées, etc.) à une unité de « comptage » dans les systèmes de traitement statistique pour améliorer les résultats. Par ailleurs, l'indexation de documents par de telles expressions pourraient affiner les recherches documentaires en tenant compte du poids sémantique formé par l'ensemble (et non seulement par ses différents constituants).

2.4.2.4 analyse syntaxique

L'analyse syntaxique consiste à regrouper des séquences de mots en syntagmes qui s'organisent autour d'un prédicat (verbe, nom ou adjectif) et à leur donner des « fonctions ». Les travaux du lexique-grammaire y ont montré le rôle fondamental du lexique, et implicitement, l'obligation d'utiliser des dictionnaires syntaxiques pour l'analyse syntaxique automatique.
E. Roche (1993, 1999) a conçu un analyseur syntaxique et transformationnel basé sur un dictionnaire syntaxique où pour chaque entrée lexicale (prédicat) est associé l'ensemble des constructions dans laquelle elle peut apparaître. Ce dictionnaire a la forme d'un transducteur qui permet de reconnaître directement les constructions dans les textes et d'étiqueter ces dernières. Le transducteur est appliqué plusieurs fois jusqu'à ce que l'analyse de la phrase ait atteint un point fixe. Le dictionnaire syntaxique a été construit semi-automatiquement à partir des tables de lexique-grammaire accumulées jusque là. A chaque table, E. Roche associe un transducteur paramétré (appelé aussi transducteur de référence ou tranducteur patron) construit manuellement. Ce dernier représente une entrée fictive de la table qui accepterait toutes les propriétés. Chaque paramètre correspond à une propriété de la table. L'objectif est d'ajuster automatiquement les paramètres en fonction du contenu de la table, pour chaque entrée. La procédure marche comme suit. Pour chaque entrée *e* de la table, on copie le transducteur de référence. Puis, pour chaque paramètre *p* trouvé, on regarde le contenu de l'élément de la table situé à l'intersection de la ligne correspondant à *e* et de la colonne correspondant à *p*. Si cet élément est + (*vrai*), alors la propriété associée est conservée dans le transducteur ; si l'élément est – (*faux*) alors la propriété associée est supprimée ; autrement, on remplace le paramètre par la valeur lexicale de l'élément. Cette procédure sera expliquée plus tard dans ce mémoire lors de l'étude de phénomènes linguistiques. Les travaux d'E. Roche servent de point de départ à une vaste entreprise de conversion totale des tables. La première étape a été réalisée par S. Paumier (2003) qui a converti une partie des tables des verbes. Il a mis au point une méthode qui génère semi-automatiquement les transducteurs paramétrés pour chaque table. Nous reviendrons sur celle-ci ultérieurement.

2.4.3 Le réseau RELEX

Le réseau RELEX est un ensemble informel de laboratoires européens travaillant dans les domaines de la linguistique et du traitement automatique des langues naturelles. Les différentes équipes travaillent sur un nombre important de langues comme le français, l'anglais, le portugais, l'allemand, l'espagnol, le norvégien, le coréen, le thaï,… Elles utilisent une méthodologie commune : le lexique-grammaire. Le lexique y occupe une place fondamentale, ce qui se traduit par la construction de bases de données linguistiques à large couverture. Les logiciels Intex et Unitex servent de plates-formes linguistiques communes pour appliquer ces ressources à des textes. Des réunions formelles sont organisées tous les ans sous la forme d'une conférence, le Colloque international « *grammaires et lexiques comparés* » et sous la forme d'un atelier de travail, les journées Intex (C. Fairon, 1999 ; A.

Dister, 2000). Par ailleurs, il existe un service de veille de corpus journalistiques sur Internet, *Glossanet*, très utile pour les linguistes (C. Fairon, 2000).

2.4.4 Réflexions et perspectives

2.4.4.1 Les apports du lexique-grammaire

Les formalismes utilisés dans le cadre du lexique grammaire sont loin d'être nouveaux dans le domaine du TAL. L'intérêt des méthodes à états finis pour l'analyse des textes a été montré il y a bien longtemps par l'équipe dirigée par Z.S. Harris à l'Université de Pennsylvanie en 1958-1959 (cf. A. Joshi et K. Hopely, 1999). Il existe par ailleurs de nombreuses équipes travaillant avec la technologie à états finis (M. Mohri, 1997 ; E. Roche et Y. Schabes, 1997 ; L. Karttunen, 2001). Par contre, M. Gross et son équipe ont mis en évidence, depuis des années, certains points linguistiques qui ressurgissent dans les travaux actuels du domaine du TAL et qui apparaissent désormais comme cruciaux. Par exemple, le rôle fondamental du lexique est reconnu par beaucoup (A. Abeillé et P. Blache, 2000) : T. Briscoe et A. Copestake (1999) notent qu'il est très difficile de trouver deux verbes pouvant entrer dans une même classe. Les mots composés sont l'objet de nouvelles études (A. Copestake et al., 2002).

Au niveau applicatif, il est de plus en plus fréquent de voir des applications dites hybrides intégrant de la linguistique aux méthodes statistiques[14]. Ces techniques ont du mal à faire leurs preuves [15] car elles utilisent des données linguistiques relativement restreintes et sommaires. Il serait intéressant d'étudier l'apport d'informations linguistiques plus fines dans les systèmes statistiques comme l'ont suggéré M. Gross et J. Senellart (1998). E. Laporte (2003) propose la constitution d'un corpus étiqueté utilisant les informations codées dans le cadre du lexique-grammaire : codage grammatical, flexionnel, mots composés, tables de lexique-grammaire utilisées, etc. Ce corpus pourrait servir de corpus d'apprentissage à des outils purement statistiques. Un tel travail permettrait d'évaluer concrètement l'apport d'informations linguistiques fines aux applications du TAL.

2.4.4.2 Les perspectives du lexique-grammaire

La levée d'ambiguïtés

L'efficacité des outils de désambiguïsation à partir de règles linguistiques locales *ad hoc* est encore à montrer. Des travaux sont en cours. Il paraît cependant clair que ces outils ne pourront à terme se passer d'une analyse syntaxique et sémantique, et probablement, au bout du processus d'analyse, de choix arbitraires d'interprétation (ex : statistiques).

Analyse de phénomènes locaux

Comme nous le verrons dans la section suivante, de nombreuses grammaires locales ont été accumulées et il reste encore beaucoup de phénomènes à décrire.

Analyse syntaxique

Deux aspects de l'analyse syntaxique méritent d'être examinés en détail : la représentation formelle du résultat et les descriptions linguistiques encore à réaliser.

D'abord, l'ambiguïté n'est pas seulement présente au niveau lexical[16], mais aussi au niveau syntaxique. En effet, la séquence *Prep GN Prep GN … Prep GN* est extrêmement ambiguë : *Prep GN* est soit un adverbe soit un argument d'un prédicat verbal, nominal ou adjectival. Le nombre d'interprétations possibles est de l'ordre de la factorielle (*n* ! avec *n* nombre de

[14] cf. F. Jelinek et al. (1992).
[15] A. Smeaton (1999), T. Strzalkowski et al. (1999), E. Voorhees (1999).
[16] Soit *n* le nombres de mots d'une phrase : le nombre d'interprétations possibles de celle-ci est d'environ 2^n.

séquences *Prep GN*)[17]. Or ce type de séquences est extrêmement fréquent dans les textes. Si l'on souhaite tenir compte de l'ambiguïté le plus loin possible dans le processus afin de ne négliger aucune piste, il est nécessaire de trouver une représentation adaptée. Le résultat non ambigu de l'analyse syntaxique est très souvent représenté sous la forme d'un arbre. Ainsi, il serait logique de représenter l'ensemble des interprétations possibles par une forêt d'arbres. Cependant, l'ambiguïté serait telle qu'une telle représentation serait trop coûteuse. Une représentation factorisée de cette forêt pourrait être un automate particulier. Ensuite, il est nécessaire de trouver des algorithmes peu coûteux associés à l'application des règles.
Au niveau linguistique, les travaux réalisés jusqu'à présent avaient pour cadre la phrase simple. Si l'on reste dans ce cadre, le comportement des constructions à prédicats nominaux lorsqu'elles sont réduites en GN devrait être examiné car les prédicats nominaux se retrouvent très fréquemment dans des groupes nominaux. Par ailleurs, les contraintes propres aux phrases complexes n'ont pas ou très peu été examinées. Des travaux sont en cours dans la suite de M. Mohri (1993).

Diffusion libre des données
Pour finir, la diffusion des données est un nouveau défi à relever. Unitex est un logiciel libre et les dictionnaires le sont partiellement. Une étape importante et obligatoire est la diffusion des bibliothèques de grammaires locales accumulées dans le cadre du réseau RELEX (cf. ce présent ouvrage).

2.5 Grammaires locales : un état des lieux

Dans cette section, nous regardons en détail un type de données linguistiques : les grammaires locales. Nous établissons un état des lieux dans le cadre du réseau RELEX. D'abord, nous décrivons formellement les grammaires locales. Puis, nous regardons leurs différents niveaux d'analyse. Enfin, nous montrons différentes applications les utilisant. Nous parlons essentiellement des travaux réalisés sur la langue française car c'est la langue qui a le niveau le plus avancé. Pour des précisions spécifiques sur d'autres langues, les lecteurs sont priés de se référer à la bibliographie générale[18].

2.5.1 Formalisme

Les grammaires que nous considérons ont la forme de réseaux récursifs de transitions (W.A. Woods, 1970, M. Silberztein, 1993). Chaque grammaire g possède un alphabet N d'éléments non terminaux, un alphabet T d'éléments terminaux (avec $N \cap T = \emptyset$), un ensemble de règles G sous la forme de graphes (terme gauche : nom du graphe, soit un élément de N ; termes droits : factorisés sous la forme d'un graphe sur $N \cup T$) et un axiome de départ (ou graphe principal) g_0. Les graphes sont construits à l'aide d'un éditeur et se lisent de gauche à droite. Les transitions étiquetées sont dans des boites. Par exemple, prenons $N = \{X,Y,Z\}$, $T = \{a,b,c,d\}$ et X correspondant au nom de g_0. G est représenté par l'ensemble des graphes ci-dessous (fig. 5, 6 et 7) où chaque étiquette grisée est un élément non-terminal (soit un appel à un sous-graphe). Ainsi, notre grammaire reconnaît des expressions telles que *aaccbb* ou *bcdab*.

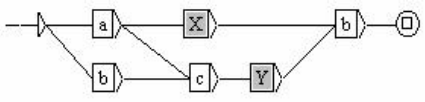

Figure 5 : X

[17] T. Briscoe et J. Caroll (1993, p. 40) font le rapprochement avec les nombres de Catalan.
[18] Site Internet www-igm.univ-mlv.fr/infolingu.

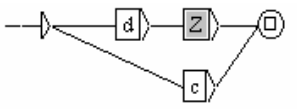

Figure 6 : Y

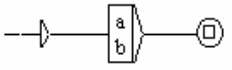

Figure 7 : Z

Pour l'instant, nous avons défini les éléments des alphabets comme de simples symboles. Nous précisons maintenant leur forme réelle dans nos grammaires linguistiques. Les symboles non-terminaux sont des noms arbitraires donnés à des graphes. Sous Intex ou Unitex, ces noms sont toujours précédés du caractère « *:* » (non représenté sur les graphes) : ce sont en quelque sorte des appels à des sous-graphes. Les symboles terminaux représentent, dans la majorité des cas, des mots au sens linguistique et sont donc très variés. Afin de limiter le nombre de transitions dans les graphes, nous utilisons des abréviations pour désigner des ensembles d'éléments terminaux. Par exemple,

- <*station*> désigne toutes les formes fléchies de la forme canonique *station* : {*station*, *stations*}

- <*V*> désigne n'importe quel verbe codé dans nos dictionnaires, c'est équivalent au OU logique de tous les verbes codés dans le dictionnaire

- <*N:ms*> désigne n'importe quel nom masculin singulier du dictionnaire (noms simples et composés)

- <*NB*> désigne n'importe quel nombre représenté par l'expression régulière *(0+1+2+3+4+5+6+7+8+9) (0+1+2+3+4+5+6+7+8+9)** où le symbole * représente le symbole de Kleene[19] et le signe + symbolise le OU logique.

Ainsi, l'exemple 8 représente une classe de mots composés sémantiquement proches de *station de ski*, et reconnaît des expressions telles que *station de sports d'hiver* ou *station de haute montagne*. Dans certains cas, l'alphabet des symboles terminaux n'est pas composé de mots linguistiques mais des caractères typographiques de la langue de travail. L'exemple 9 représente les différentes variantes orthographiques du toponyme *Vietnam* (O. Piton et D. Maurel, 1997). Notons également qu'un mélange des deux niveaux est possible, notamment pour des formes telles que *re#*<*V*> où # est un symbole qui interdit l'espace entre le préfixe *re* et le verbe <*V*>.

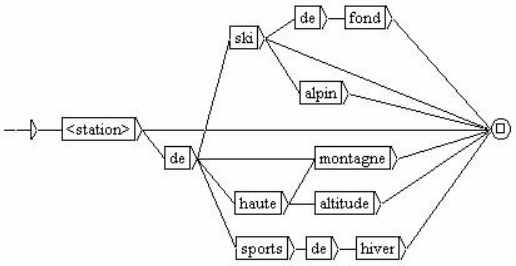

Figure 8 : Station

[19] cf. T. Sudkamp (1997) pour des détails précis sur les expressions régulières. Pour information, *a** reconnaît des suites de 0 ou plus d'éléments *a*. L'élément vide est donc reconnu.

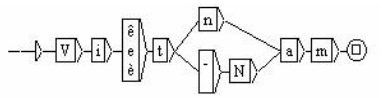

Figure 9 : Vietnam

Il est possible d'ajouter des informations en sortie des graphes. Ainsi, nos grammaires peuvent se comporter comme des transducteurs à états finis (J. Berstel, 1979 ; M. Silberztein, 1999). Par exemple, le graphe 10 qui décrit des adverbes de temps tels que *à l'aube* ou *en fin de matinée* peut servir à étiqueter les expressions qu'il reconnaît comme des adverbes de temps à l'aide des informations de sortie écrites en gras sous les boites du graphe. Lorsqu'elles ne sont pas représentées, les sorties sont vides. Ainsi, après application de cette grammaire, le texte *Marie est arrivée en fin de matinée* peut être étiqueté *Marie est arrivée <ADV+Time> en fin de matinée<\ADV+Time>*.

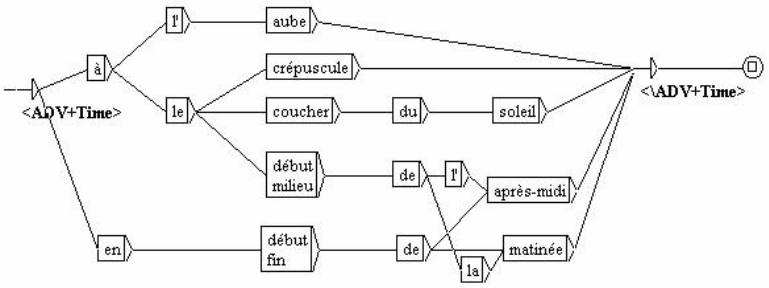

Figure 10 : Adverbes de temps

Par ailleurs, il est possible de construire des graphes décrivant des règles de réécriture : ce sont des graphes à variables. Le graphe 11 permet de décrire la phrase *Paul est à 10 km de la ville* et de l'interpréter sémantiquement : la distance *d* entre *Paul* et la *ville* est égale à *10 km* ou *d(Paul,la ville)=10 km*. En effet, les séquences reconnues par les morceaux de graphes entre parenthèses indexés par l'identifiant *i* (*i* : entier) sont stockées dans les variables *$i* et sont réécrites en sortie de l'application du graphe quand elles apparaissent en sortie dans le transducteur.

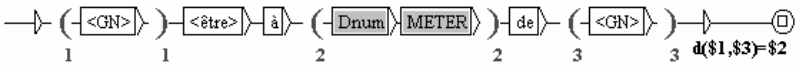

Figure 11 : Interprétation sémantique

Enfin, les graphes-patrons représentent des graphes paramétrés qui décrivent le méta-ensemble des structures dans lesquels peuvent rentrer les entrées lexicales d'une table (E. Roche, 1993 ; J. Senellart, 1999a). Ils permettent de transformer automatiquement les informations codées sous la forme de tables de lexique-grammaire en graphes. Pour chaque entrée lexicale (ou chaque ligne), on crée automatiquement un graphe associé à partir des informations de la table. Nous utilisons un système de variables qui sont placées dans les boites des graphes. Soient *i* et *j* deux entiers, *TLG* une table de lexique-grammaire et *g* le graphe-patron associé à *TLG*. Etant donné une ligne *i* de *TLG*, la variable *@j* (se trouvant dans *g*) correspond au contenu de l'intersection de la ligne *i* et de la colonne *j* de *TLG*. Ainsi, chaque variable correspond à une colonne des tables, soit une propriété, c'est-à-dire un ensemble de séquences ou/et des informations lexicales. Pour chaque ligne *i*, si *TLG(i,j)* = -, alors on supprime la boite contenant *@j*, on supprime un ensemble de séquences non désirées. Si *TLG(i,j)* = +, on remplace *@j* par l'élément vide. Par défaut, on remplace *@j* par le

contenu de *TLG(i,j)*. Par exemple, à partir de la table 12 et de son graphe-patron associé (figure 13), on génère automatiquement les deux graphes de la figure 14.

C1	C2
X	+
Y	-

Figure 12 : Table

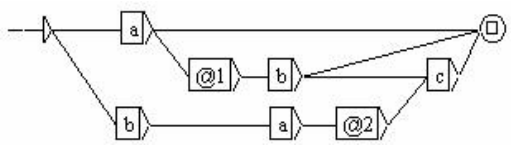

Figure 13 : Graphe de référence

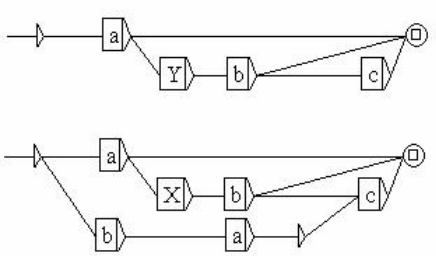

Figure 14 : Graphes générés à partir de la table 12 et du graphe 13

2.5.2 Les différents niveaux d'analyse

Le gros avantage des grammaires sous la forme de graphes est qu'elles permettent différents niveaux d'analyse des textes. Dans la précédente section, nous avons déjà distingué deux niveaux selon l'unité minimale utilisée (caractère ou mot). Lorsque l'unité minimale est le caractère, nous pouvons parler de traitement morphologique. Dans ce cas, les graphes utilisés servent à décrire des variantes orthographiques de manière compacte et donc à alimenter les dictionnaires électroniques. Nous nous intéressons maintenant au cas où l'unité minimale est le mot. Les niveaux d'analyse y sont plus nombreux. Tout d'abord, les graphes peuvent être assimilés à des extensions des dictionnaires des mots composés. Par exemple, la description des dates sous la forme d'automates factorise de façon significative un ensemble d'expressions quasiment impossible à traiter sous forme de listes (D. Maurel, 1990 ; J. Baptista, 1999). Ensuite, il est possible de décrire les contraintes locales autour d'un mot de manière très fine. Ainsi, nous pouvons constituer des classes de mots composés ayant un sens proche, comme le graphe **Station** (fig. 8). Ce dernier graphe est un moyen de représenter explicitement une des deux entrées lexicales de *station* : (1) *station (E + de ski)* et (2) *station (E + de métro)*. J. Senellart (1998) a construit des grammaires pour des noms d'activités telles que *ministre (E + de l'intérieur)*. L'étape suivante est de construire des constituants de phrases comme les groupes nominaux ou les groupes verbaux comme le montre M. Salkoff (1973) pour construire une grammaire en chaîne du français. Afin de reconnaître automatiquement des expressions figées dans les corpus, J. Senellart (1999b) a élaboré quelques grammaires de groupes nominaux simples comme montré dans le graphe 15 ci-dessous[20]. C. Dominguès (2001) a regardé le comportement de groupes nominaux contenant une coordination. La constitution de grammaires complètes reconnaissant les GN est l'un des futurs enjeux du réseau RELEX. Par ailleurs, il existe des grammaires de groupes verbaux

[20] Le symbole <*E*> désigne le mot vide.

composés en anglais (M. Gross 1999) et en français (M. Constant et al., 2002 ; S. Paumier, 2003).

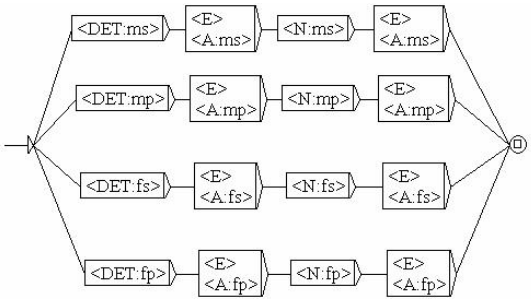

Figure 15 : GN

A partir de l'étape précédente, il est possible de décrire des phrases simples libres contenant un prédicat (verbe, nom, adjectif) et des arguments comme l'a fait E. Roche (1993, 1999) à l'aide de tables de lexique-grammaire et de transducteurs à états finis. Un travail de grande envergure dans la continuité de cette étude est actuellement mené au sein de l'université de Marne-la-Vallée. J. Senellart (1999a) a décrit, à l'aide de graphes, les expressions figées du français, à partir des tables de M. Gross (1984) et montré par la même occasion leur présence en grand nombre dans les textes français (à peu près dans une phrase sur trois dans des textes journalistiques). Les phrases simples avec prédicats nominaux et adjectivaux peuvent être réduites en groupe nominal (GN) ou groupe adjectival. Leur description permet d'affiner celle des GN généraux. La reconnaissance de phrases complexes, combinaisons de phrases simples sous toutes les formes, d'adverbes de temps (M. Gross, 2002), de lieux (M. Constant, 2002b), d'incises (C. Fairon, 2000) et de conjonctions, est un objectif majeur du réseau RELEX. Enfin, quelques études ont été menées dans le domaine spécifique de la bourse (M. Gross, 1997 ; T. Nakamura, à paraître) et ont montré la limitation du lexique et des structures syntaxiques employés. Des grammaires entièrement lexicalisées ont ainsi pu être construites.

2.5.3 Les applications

Nous décrivons, dans cette section, quelques applications qui ont été développées par les membres de la communauté RELEX. Elles sont nombreuses ; nous listons les principales :

(découpage) Une des premières étapes du traitement automatique des textes est la segmentation de ces derniers en phrases à l'aide de la ponctuation. Des transducteurs ont été construits à cet effet, insérant à la fin des phrases un symbole comme *{S}*(N. Friburger et al., 2000). Les résultats sont encourageants et dépendent beaucoup des corpus.

(morphologie) Un problème crucial dans la construction de dictionnaires à large couverture est la génération automatique de toutes les formes fléchies d'un lemme. A chaque lemme est associé une classe de flexion représentée par un transducteur. L'application de ce transducteur permet de résoudre une grande majorité des problèmes rencontrés (M. Silberztein, 1997).

(étiquetage) La consultation des dictionnaires permet de faire un étiquetage lexical des textes. Le résultat est représenté sous la forme d'un transducteur permettant par la même occasion de montré l'impressionnante ambiguïté de la langue (M. Silberztein, 1997). Les grammaires lexicalisées permettent de reconnaître des séquences composées et d'étiqueter ces dernières de manière très satisfaisante. La prochaine étape est l'analyse syntaxique complète de textes. Pour cela, il est nécessaire de trouver, en tenant compte de l'énorme ambiguïté de la langue,

de nouveaux formalismes et des algorithmes associés. Des travaux sont en cours à l'université de Marne-la-Vallée.

(extraction) Un des sujets à la mode actuellement est l'extraction d'information. Le plus brûlant d'entre eux est l'extraction de noms propres. De nombreuses études ont été menées et sont en cours. J. Senellart (1998) extrait automatiquement des noms de personnalités en leur associant une fonction politique ou professionnelle à l'aide de grammaires sous forme de graphes. N. Friburger et al. (2001) extraient des noms propres de personne à l'aide de cascades de transducteurs. D'autres sujets ont aussi été abordés comme l'extraction de noms de gènes dans les corpus en génomique (T. Poibeau, 2001). Une application directe de l'extraction sont les systèmes de questions-réponses automatiques (par exemple, C. Fairon et P. Waitrin, 2003).

(filtrage) Le filtrage d'information est aussi un sujet majeur de ces dernières années, notamment pour la distribution personnalisée des dépêches AFP. A. Balvet (2000) montre l'intérêt d'utiliser des graphes linguistiques pour repérer les textes adéquats à une requête donnée.

(ambiguïté) La levée d'ambiguïté des textes est fondamentale pour le traitement automatique. De nombreuses études pointues ont montré l'intérêt d'utiliser des batteries de transducteurs (M. Silberztein, 1997 ; A. Dister, 1999 ; P. Carvalho et al., 2002, 2003). C'est un sujet très important dans la communauté : un module de levée d'ambiguïté (ELAG) a même été implémenté par E. Laporte et al. (1999) qui permet de supprimer les mauvais chemins dans le transducteur du texte.

(traduction) La traduction automatique des langues est sûrement l'objectif le plus difficile du TAL (M.Gross, 1992). Devant la difficulté de la tâche, les études dans ce domaine s'orientent vers l'aide automatique à la traduction. C. Fairon et J. Senellart (1999) ont construit un ensemble de transducteurs lexicalisés traduisant des adverbes de temps du français à l'anglais. Dans le même esprit, J. Baptista et D. Catala (2002) ont réalisé quelques grammaires permettant de traduire des adverbes de temps du portugais vers l'espagnol et vice-versa.

(génération) La génération n'est pas un sujet très porteur dans la communauté RELEX plus attirée par l'analyse. Cependant, il existe un projet de génération automatique de sujets d'examen à l'aide de graphes à variables ou graphes de réécriture (C. Fairon, 2001).

2.5.4 Quelques remarques

Comme l'indique leur nom, les grammaires locales représentent des phénomènes locaux, dans la très grande majorité des cas, indépendants du reste de la phrase. Elles sont donc très intéressantes car leur construction ne dépend pas ou très peu d'autres phénomènes linguistiques, ce qui est le cas dès que l'on veut faire une étude sur l'analyse d'un phénomène dans l'analyse syntaxique globale.
L'accumulation et l'hétérogénéité des grammaires locales (cf. section précédente) rendent nécessaire la mise en place de méthodes claires et précises de construction afin de faciliter la collaboration entre les équipes et au sein même des équipes. Les deux prochains chapitres mettent à plat un ensemble de méthodes à l'aide d'exemples originaux. Nous en proposons quelques unes qui nous semblent simples et raisonnables. Toute construction de grammaire lexicalisée nécessite auparavant une analyse linguistique fine et systématique des phénomènes locaux en fonction du lexique utilisé comme le montre M. Gross (1997). Cette première étape

franchie, il existe deux méthodes de codage suivant la taille du lexique employé et les variations lexico-syntaxiques des séquences décrites :

- Soit directement à l'aide d'un éditeur de graphes ;
- Soit au moyen d'une représentation temporaire : des tables de lexique-grammaire qui sont ensuite transformées semi-automatiquement en grammaires locales.

La construction directe de grammaires locales est privilégiée car leur lecture et leur compréhension sont immédiates. Par contre, dès que le nombre de structures augmente, les graphes ont tendance a rapidement devenir illisibles, si l'auteur n'a pas une organisation rigoureuse. Les tables de lexique-grammaire demandent un temps d'adaptation pour comprendre leur mécanisme, ce qui peut être vu comme un frein au premier abord[21]. Cependant, elles permettent de représenter clairement de grands ensembles de structures syntaxiques. Ce type de représentation est utile pour éviter la répétitivité des tâches comme nous le verrons ultérieurement. Le choix d'une ou l'autre des deux méthodes dépend essentiellement du goût du linguiste, même si, dans certains cas, le choix est évident pour tous. Notre démarche étant essentiellement empirique, nous souhaitons avant tout donner des exemples concrets et originaux en n'entrant pas dans un débat théorique qui ne cadrerait pas avec notre travail.

Les grammaires locales sont appelées à moyen terme à être appliquées sur l'automate du texte (ou une structure tenant compte de l'ambiguïté de la langue). Leur application provoquera du bruit, surtout quand les séquences reconnues sont courtes. Par exemple, *le samedi* est soit un adverbe (reconnu par une grammaire locale lexicalisée de dates) soit un groupe nominal :

Le samedi est un jour de repos
Le samedi, Max chante dans une chorale

Elles ont l'avantage que chaque élément des séquences reconnues peut être désambiguïsé en interne. Par exemple, si l'on reconnaît *à le début de la soirée* par une grammaire de date, on est sûr, dans le cas où la séquence est un adverbe de date que *à* est une préposition, *le* un déterminant, *début* un nom, etc. A terme, il sera donc nécessaire d'ajouter des informations internes dans les grammaires pour permettre cette levée d'ambiguïté conditionnelle[22], comme montré dans le graphe ci-dessous :

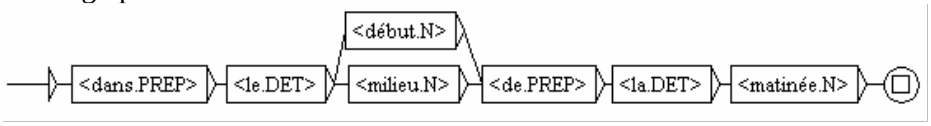

Figure 16 : informations internes

Dans ce mémoire, nous commençons par un exemple simple que nous qualifierons de standard dans le cadre du lexique-grammaire : les expressions de mesure. Nous verrons que même des cas aussi simples nécessitent une attention particulière et des descriptions fines. Puis, nous regardons un cas linguistique un peu plus complexe : les adverbes locatifs. Nous montrons le processus complet nécessaire à leur analyse linguistique et à leur représentation informatique. Nous verrons que certaines formes nécessitent des représentations un peu plus évoluées que celles utilisées traditionnellement au sein du lexique-grammaire : des systèmes de tables relationnelles. Nous présenterons les extensions formelles et algorithmiques qu'induisent ces phénomènes linguistiques.

Les méthodes que nous décrirons permettent de construire facilement et rapidement de nombreuses grammaires locales. La perpective d'une explosion de leur nombre nous incite

[21] Tout est relatif. La complexité du formalisme HPSG par exemple est nettement plus importante.

[22] Si, après analyse complète de la phrase, l'analyse de la grammaire locale est conservée alors les éléments internes sont désambiguïsés.

d’ailleurs, dans une dernière partie, à élaborer des outils de gestion d’une bibliothèque de grammaires locales.

Chapitre 3 : Analyse et représentation d'expressions de mesure

3.1 Introduction

Dans cette partie, nous décrivons et représentons des expressions numériques et plus particulièrement des expressions de mesure[23], dans le but de les reconnaître automatiquement dans les textes de langue générale (quotidiens d'informations comme *Le Monde* et journaux de vulgarisation scientifique type *Science et Vie*)[24]. De précédentes versions de ce travail ont été publiées dans M. Constant (2000, 2002a).
Nous employons le terme expression de mesure pour toute séquence linguistique contenant la sous-séquence de base *Dnum Unité* (ex : *10 m*) où *Dnum* est un déterminant numérique cardinal. Nous distinguons deux types de mesure, les mesures « absolues » et les mesures « relatives » qui entrent respectivement dans les structures suivantes où le symbole *Ng* désigne un nom de grandeur :

(1) *N*0 *avoir un Ng de Dnum Unité* = : *la corde a une longueur de 10 m*
(2) *N*0 *Vsup Prép un Ng de Dnum Unité Prép*1 *N*1[25] = : *le couteau forme un angle de 10° avec la fourchette*

Les mesures absolues sont des mesures de caractéristiques ou propriétés (*Ng*) propres à l'argument *N*0, comme la taille ou la durée :

Max a une taille de 1,71 m
Le spectacle a une durée de trente minutes

Les mesures relatives mesurent une caractéristique ou propriété (*Ng*) de *N*0 par rapport à *N*1, comme la distance

[23] Des outils statistiques d'extraction d'expressions de mesures existent (R. Agrawal et R. Srikant, 2002).
[24] Nous n'étudions pas les discours techniques.
[25] *Prép* peut être vide.

Paris est à une distance de 600 km de Bordeaux

Nous intégrons également les expressions de pourcentage dans cette dernière catégorie, qui mesure l'inclusion d'un ensemble (*N*0) dans un autre (*N*1) :

Les étudiants représentent 10% de la population

Nous faisons aussi quelques remarques sur des séquences qui expriment une comparaison « relative » telles que dans la phrase suivante :

Max est deux centimètres plus grand que Luc

Dans cette section, nous décrivons en détail le comportement syntaxique de ces expressions dans des phrases élémentaires, puis dans les formes réduites de ces phrases. Nous en donnons des représentations simples sous la forme de grammaires locales et de tables syntaxiques. Cette étude a été réalisée sur le français et partiellement sur l'anglais. Nous attachons une grande importance à la réalité linguistique du contenu des textes. Notre travail est basé sur les études de M. Silberztein (1993) et A. Chrobot (2000) sur les déterminants numériques, de J. Giry-Schneider (1991) sur les phrases élémentaires représentant une mesure absolue[26] et de P. A. Buvet (1993, 1994) sur les déterminants nominaux.

3.2 Les composants élémentaires

3.2.1 Généralités

Nous avons défini une expression de mesure comme une séquence comportant la séquence *Dnum Unité* où *Dnum* désigne un déterminant numérique et *Unité* une unité de mesure. Nous souhaitons, dans un premier temps, décrire, de manière détaillée, chaque composant simple de cette séquence. Les déterminants numériques sont les plus variés et nous utilisons une typologie formelle. Il peut s'agir de :

- déterminants indéfinis au pluriel (*Dind-pl*) comme *des*, *plusieurs*, *quelques*, etc.
- déterminants numériques cardinaux simples ou composés écrits en lettres (ex : *douze* ; *quarante-trois*)
- séquences de chiffres arabes décrivant des nombres réels dans un format standard (ex : *1 905* ; *1,78*) ou dans un format scientifique (ex : *1,54.10+5*).
- déterminants nominaux de la forme *Det Nnum de* (=: *des milliers de*) où *Nnum* est un nom que l'on qualifiera de numérique comme *milliers*, *dizaines*, etc.

Nous étudions spécifiquement les trois derniers types. Après avoir examiné les différentes classes d'unités simples que nous utilisons, nous évoquons le cas des prédéterminants numériques qui sont essentiels pour une reconnaissance fine des mesures car ils introduisent de légères modifications sémantiques (ex : *à peu près dix ampères* ≠ *exactement dix ampères*).

Nous analysons également le schéma de phrase *Det Ng être de Dnum Unité* (=: *la longueur est de 30 m*) qui est commun aux deux structures de mesure que nous allons étudier. Cette étude va nous permettre de construire des graphes élémentaires de mesure à partir des graphes d'unités.

[26] Des travaux ont aussi été réalisés par A. Borillo (1985, 1998).

3.2.2 Graphes des déterminants numériques

3.2.2.1 Les déterminants numériques cardinaux écrits en lettres

Nous traitons brièvement ce point car les déterminants numériques cardinaux écrits en lettres ont déjà été étudiés et décrits sous la forme de graphes par M. Silberztein (1993) pour le français et A. Chrobot (2000) pour l'anglais. Dans cette section, nous reprenons les points importants de l'étude sur le français. Nous notons *DnumEnLettres* ce type de déterminants numériques, i.e. les nombres entiers[27] écrits en toutes lettres (borne supérieure : *un billiard*). L'utilisation des sous-graphes a un avantage indéniable car certaines séquences peuvent apparaître plusieurs fois dans un nombre (ex : *douze* dans *douze cent douze*). Certains termes comme *cent*, *mille*, *quatre-vingts* posent quelques problèmes orthographiques car :

- *mille* est invariable ;
- *cent* est au pluriel lorsqu'il est multiplié : *deux cents* ; mais il reste invariable lorsqu'il est suivi d'un autre nombre ou qu'il est utilisé comme centième [Larousse, 2002] : *trois cent vingt, deux mille deux cent* ;
- *Quatre-vingt* est au singulier lorsqu'il est suivi d'un nombre : *deux cent quatre-vingt-deux* ; mais il est au pluriel autrement : *mille quatre-vingts*. [Grevisse, 1975, paragraphe 406]

Ainsi, pour chaque type de nombres entiers (nombres inférieurs à cent, à mille, à un million, etc.), il est nécessaire de construire deux graphes : l'un décrivant ces nombres lorsqu'ils se trouvent dans la partie droite (ou finale) d'un nombre et l'autre décrivant ces nombres lorsqu'ils sont dans la partie gauche. Par exemple, *quatre-vingt(s)* a deux comportements suivant qu'il est à droite (*quatre-vingts*) ou à gauche (*quatre-vingt*) comme dans *quatre-vingt mille deux cent quatre-vingts*.
M. Silberztein (1993) n'a pas décrit les nombres se terminant par *million(s)* et *milliard(s)*. Ces nombres sont suivis de la préposition *de* : *cent vingt millions de*. Ce type de nombres rentre donc dans une structure différente : celle des déterminants nominaux (*Det N de*) décrits dans M. Gross (1986). Notons que nous n'avons pas traité le cas du décimal *un demi* suivi d'un tiret (ex : *une demi-heure*).

3.2.2.2 Nombres écrits en chiffres arabes

Nous notons ce type de déterminant *DnumEnChiffres*. Les nombres écrits sous la forme d'une suite de chiffres ont une syntaxe bien particulière qui diffère en français et en anglais. Une solution simple et naïve pour décrire les entiers naturels est de les représenter comme une suite de chiffres soudés d'au moins un élément. Cependant, cette représentation est trop simpliste. Les nombres avec plus de trois chiffres ne rentrent pas dans ce schéma : par exemple, en français, dans *1 298*, il existe un espace blanc obligatoire[28] entre le troisième chiffre en partant de la droite et le dernier chiffre à gauche ; en anglais, l'espace blanc est remplacé par une virgule (*1,298*). D'une manière générale, un espace blanc (une virgule en anglais) apparaît obligatoirement dans la séquence de chiffres tous les trois chiffres en partant de la droite. En français, l'ensemble des entiers naturels écrits en chiffres arabes peut donc être représenté par le graphe **NombreEntierEnChiffres** ci-dessous. Le graphe **3Chiffres** décrit une suite de trois chiffres soudés et le graphe **Chiffre** l'ensemble des chiffres arabes. Le symbole # indique que l'élément à sa gauche et celui à sa droite sont soudés l'un à l'autre ; autrement dit, tout espace blanc est interdit entre ces deux éléments. Cette description précise permet de lever, dans certains cas, l'ambiguïté qui existe naturellement entre un déterminant numérique et une date désignant une année écrite en chiffres. En effet, ce type de date est une

[27] Les nombres décimaux ne semblent pas pouvoir être écrits en toutes lettres.
[28] Parfois, c'est un point qui apparaît à la place de l'espace.

suite de chiffres collés. Ainsi, *2003* ne sera pas reconnu comme un déterminant numérique par notre grammaire. Par ailleurs, il est usuel d'utiliser des nombres décimaux écrits en chiffres. En français, la partie entière est séparée de la partie décimale par une virgule (*12,7* ou *3,896*). En anglais, c'est un point qui fait office de séparateur (*12.7* ou *3.896*). Dans ce cas, la partie décimale est une simple séquence de chiffres collés d'au moins un élément et est représentée par le graphe **PartieDecimale**. Un nombre quelconque écrit en chiffres arabes est alors reconnu par le graphe **DnumEnChiffres** regroupant les graphes **NombreEntierEnChiffres**[29] et **PartieDecimale**. Notons que ces deux parties (entière et décimale) sont obligatoirement collées à la virgule les séparant. Cela évite, par exemple, de reconnaître *10, 11* dans l'expression coordonnée *10, 11 ou 12 chaises*. Par ailleurs, les nombres peuvent être signés : ils peuvent avoir un + ou un – placé au début (à gauche).

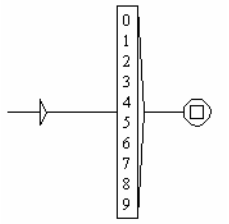

Figure 17 : Chiffre

Figure 18 : 3Chiffres

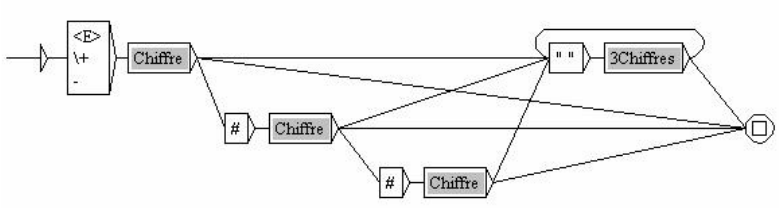

Figure 19 : NombreEntierEnChiffres

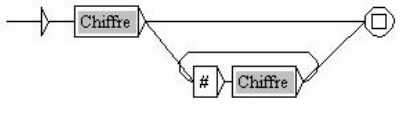

Figure 20 : PartieDecimale

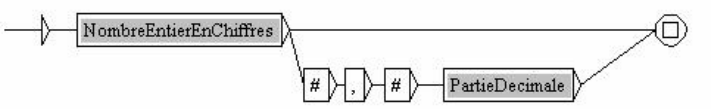

Figure 21 : DnumEnChiffres

Les nombres en notation scientifique possèdent aussi une syntaxe bien particulière. La partie entière ne comprend qu'un seul chiffre. La partie décimale est plus libre en fonction de la précision que l'on souhaite avoir. On ajuste ensuite ce nombre à l'aide d'une puissance de 10 (soit négative, soit positive) :

1,23.10E5 ou 1,23x10+5[30]
4,8x10-6 (dans *Science et Vie*)

Nous représentons de tels nombres dans le graphe **FormuleScientifique** ci-dessous :

[29] Le symbole \ devant + est un symbole de déspécialisation. Le symbole <*E*> est le mot vide.
[30] Nous ne regardons que des textes bruts sans tenir compte de l'enrichissement typographique : 1,23.10-6 serait plutôt écrit $1{,}23.10^{-6}$.

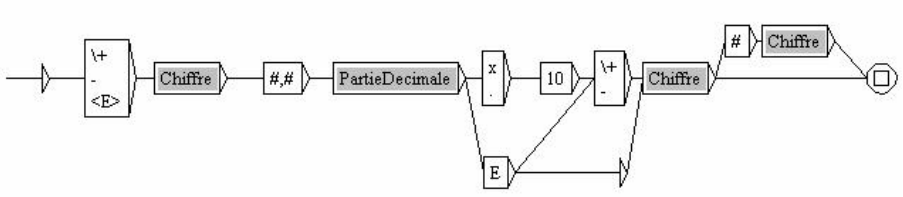

Figure 22 : FormuleScientifique

3.2.2.3 Les déterminants nominaux numériques

Nous regardons maintenant les déterminants nominaux de la forme *Det Nnum de* où *Nnum* est un nom que l'on qualifiera de numérique, tel que *milliers, millions, milliards* :

Dix millions de personnes sont allergiques à la poussière

Notons que le déterminant numérique *dix* avant *millions* peut aussi s'écrire en chiffres. *Det* peut également être un déterminant indéfini pluriel :

10 millions de personnes sont allergiques à la poussière
Des millions de personnes sont allergiques à la poussière

Il est également possible d'utiliser le modifieur numérique *demi* :

Un demi million de personnes sont allergiques à la poussière

Nous définissons les *Nnum* à partir de sa possibilité d'occurrence dans :

- des multiples exacts de 10 : *million(s), milliard(s), billion(s)*

  Cette planète est à 17 milliards d'années-lumière de la Terre

- des sous-multiples[31] de 10 : *dixième(s), centième(s), milliardième(s)*

  Ce robot a une précision d'un millionième de centimètre

- d'autres termes : *millier(s), centaine(s), cinquantaine(s), douzaine(s), dizaine(s)*, etc.

  Marie attend une vingtaine d'amis cette semaine

Il est possible d'insérer un ensemble restreint de modifieurs dans notre séquence comme dans :

Paul a perdu une (E + bonne + petite) dizaine de kilos

Par ailleurs, on peut combiner les structures *Det Nnum de* afin de former des structures plus complexes de la forme *Det Nnum de (Nnum de)**. C'est le cas dans les exemples ci-dessous :

1,67 milliardième de milliardième de milliardième de kg. (revue *Science et Vie*)

[31] Il est également possible d'utiliser des fractions du type *deux tiers* :
Ce robot a une précision de deux tiers de centimètre

La dernière phrase ci-dessus montre que l'on peut combiner différents types de *Nnum*. D'autre part, il existe des variantes semi-figées de ces séquences qui ont la structure *des N et des N de* :

Marie a gagné des centaines et des centaines d'amis dans cette affaire.

Notons que la contraction de la préposition *des* en *de* est également possible :

La galaxie est constituée de millions et de millions d'étoiles.

Les deux noms de la structure doivent être identiques et les déterminants sont obligatoirement *des* :

* *Léa a acheté des dizaines et des centaines de chiens*
* *Léa a acheté plusieurs dizaines et plusieurs dizaines de chiens*

Nous avons rassemblé toutes ces expressions dans le graphe **DetNnumDe**[32] :

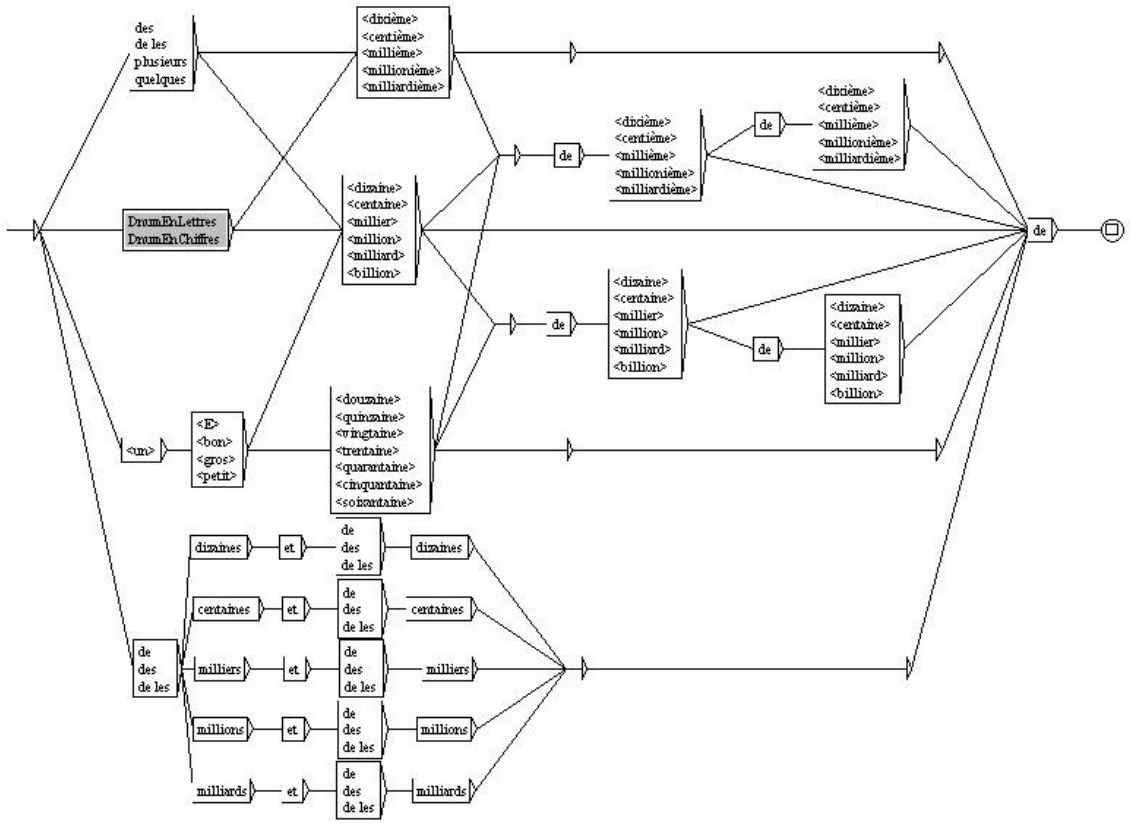

Figure 23 : DetNnumDe

[32] Ce graphe n'est pas cyclique contrairement à ce que l'on aurait pu penser. En effet, nous décidons de limiter le nombre de répétitions de la séquence *Nnum de* car les séquences trop longues sont difficilement compréhensibles par les lecteurs. Par ailleurs, certaines informations comme les fractions ne sont pas représentées.

Remarque générale sur les nombres :

Nous regroupons tous les graphes représentant des déterminants numériques dans le graphe **Dnum**.

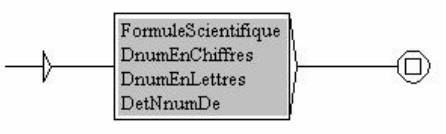

Figure 24 : Dnum

Il existe des combinaisons de ces déterminants numériques formant, par exemple, des approximations sous la forme d'intervalles (*entre 2 et 3 mètres* ; *de 7 à 9 kg*). Cependant, comme nous le verrons ultérieurement leur comportement ne peut être étudié de manière locale, mais dans le cadre d'une phrase élémentaire.

3.2.2.4 Les prédéterminants numériques

Dans les textes, on constate la présence de prédéterminants numériques (M. Gross, 1977) qui se trouvent avant ou après la séquence *Dnum N* (dans notre cas, *Dnum Unité*) comme *presque, environ, exactement, à peu près*:

Il y a 45 enfants environ
Max a mangé à peu près 30 fruits
Marie a (presque + environ + exactement) 10 ans
Marie a 10 ans (environ + très exactement)

Ces mots modifient l'interprétation de la valeur du déterminant numérique. La distribution des prédéterminants situés avant la séquence *Dnum N* (*PreDnum*) et des prédéterminants situés après la séquence *Dnum N* (*PreDnumPost*) n'est pas la même :

*Luc possède (à peu près + environ + *ou presque + presque) 30 voiliers*
*Luc possède 30 voiliers (?à peu près + environ + ou presque+*presque)*

Les prédéterminants *PreDnumPost* ne peuvent apparaître entre le déterminant et le nom :

** Il y a 45 (environ + très exactement) enfants*
** Marie a 10 (environ + très exactement) ans*

Cette contrainte n'est cependant pas toujours vrai comme le montre la phrase suivante (pas très naturelle) :

? Il y a quelques dizaines environ de voitures à boites automatiques[33]

Il est possible d'avoir à la fois un prédéterminant *PreDnum* et un prédéterminant *PreDnumPost* comme dans la phrase :

Paul a couché avec environ mille femmes au total

[33] Le symbole *?* signifie que la phrase qu'il précède n'est pas très naturelle.

Il est très facile de représenter ces deux ensembles sous la forme de graphes comme montré ci-dessous[34] (graphes **PreDnum** et **PreDnumPost**), puis de les incorporer dans notre structure de base.

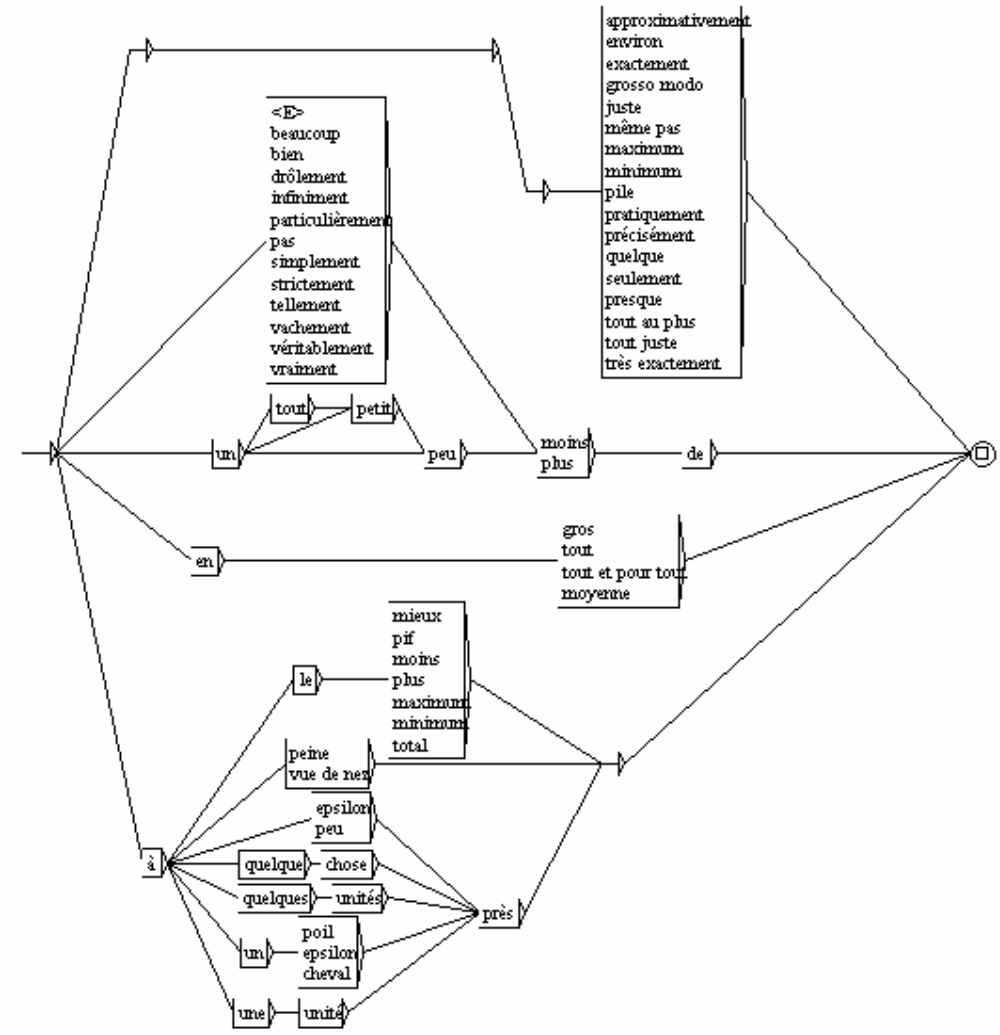

Figure 25 : PreDnum

[34] La base de ces graphes nous a été fournie par M. Gross.

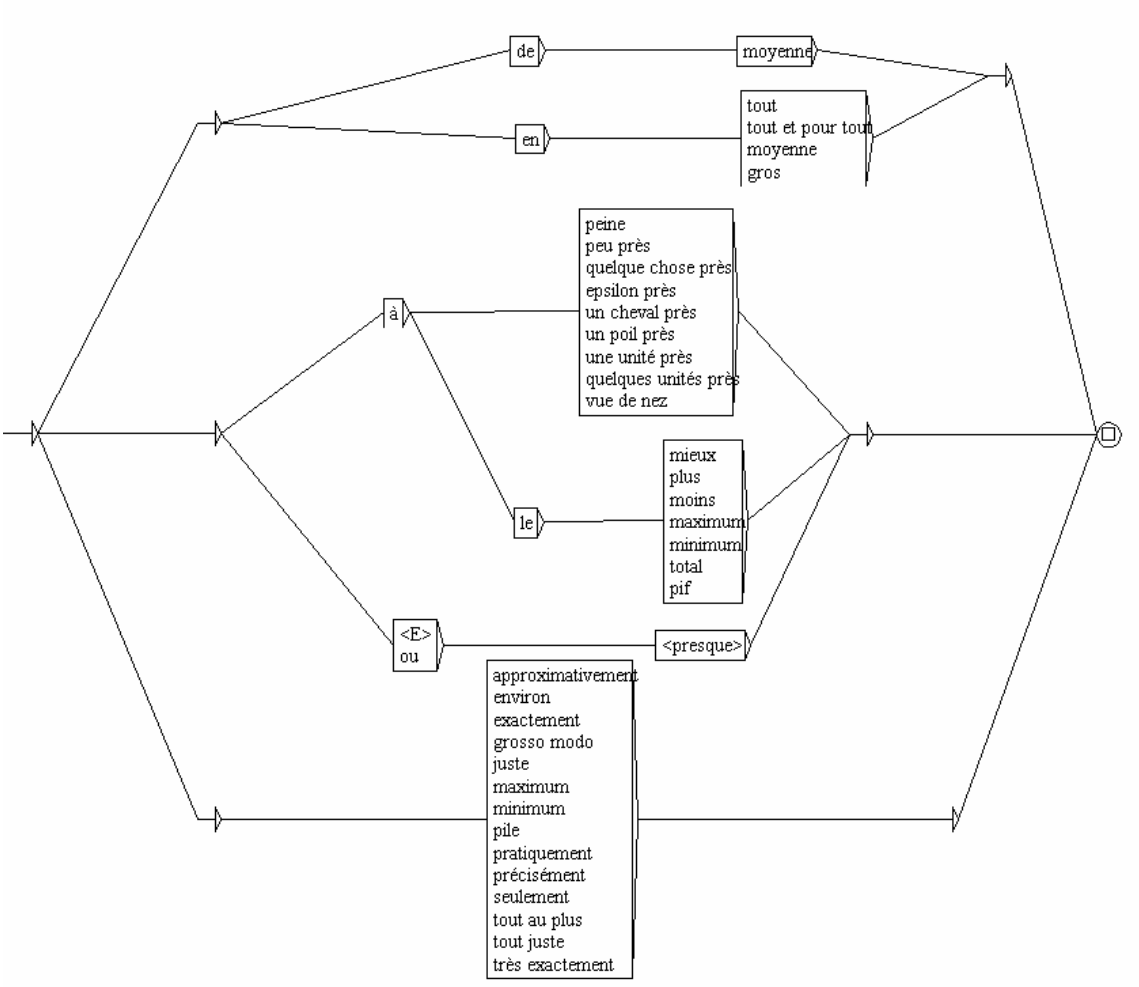

Figure 26 : PreDnumPost

Cependant, le comportement syntaxique des prédéterminants est plus complexe que cela et ils ne peuvent être uniquement étudiés de manière locale. Il faut les considérer dans des phrases simples comme dans M. Gross (1977). Par exemple, ils peuvent jouer le rôle d'adverbes car certains peuvent s'insérer n'importe où dans les phrases :

Max a (au total + à un poil près + ?environ) dépensé 400 euros
*(Au total + A un poil près + *Environ), Max a dépensé 400 euros*

Nous reviendrons sur ce phénomène ultérieurement lorsque nous examinerons nos différents schémas de phrase représentant des mesures.

3.2.3 Graphes élémentaires de mesure

3.2.3.1 Les unités

Dans cette section, nous répertorions les unités de mesure. Nous utilisons la classification scientifique : étant donné une unité de base (*mètre* ; *m*), nous regroupons, dans une même classe (ou graphe), ses multiples et sous-multiples (ex : *kilomètre* ; *km* ; *millimètre* ; *mm*). Nous divisons chaque classe en deux : les unités écrites en toutes lettres (*millimètre*,

centimètre) et les symboles des unités (*mm*, *cm*). Nous donnons ci-dessous les graphes **Metre** et **Metre_abr**. Chaque unité écrite en toutes lettres est mise entre angles : par exemple, *<millimètre>* est l'ensemble {*millimètre*, *millimètres*}. Cela signifie que l'on considère que les unités ont été décrites dans les dictionnaires que nous allons utiliser pour appliquer nos grammaires. Les symboles sont, quant à eux, écrits entre guillemets car ils doivent être reconnus tels quels dans les textes : la séquence *"cm"* interdit les variantes en majuscules, c'est-à-dire *CM*, *cM*, *Cm*.

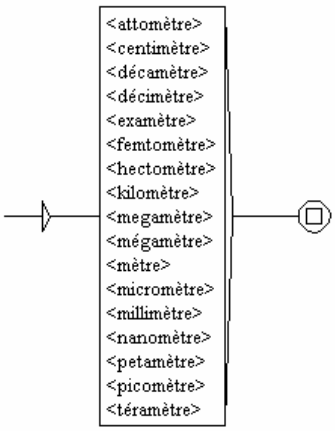

Figure 27 : Metre

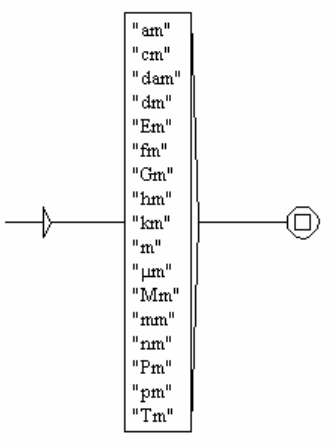

Figure 28 : Metre_abr

Nous répertorions ci-dessous les 20 graphes que nous avons construits à l'aide du dictionnaire *Larousse* :

> Ampere, Are, Bit, Calorie, DegreCelsius, ElectronVolt, Gramme, Hertz, Joule, Kelvin, Litre, Livre, Metre, Mile, Mille, Newton, Octet, Seconde, Volt, Tonne

A chacun de ces graphes, nous associons le graphe des symboles des unités correspondants dont le nom se termine par *_abr*. La plupart des graphes ne présentent aucune difficulté et leur contenu est facilement construit manuellement (voire automatiquement) sur le même modèle que nos deux exemples.
Le graphe **DegreCelsius** décrit les différents types de degrés pour mesurer une température : *degré Celsius*, *degré Fahrenheit*, *degré Kelvin*. Les symboles décrits dans **DegreCelsius_abr** sont *°C*, *°F* et *°K*. Le graphe **Mille** correspond aux milles nautiques. Il n'existe pas de graphe *_abr* associé. Nous avons considéré que les unités de temps n'étaient pas suffisamment bien représentées par le graphe **Seconde** (*seconde*, *milliseconde*, …). Nous avons donc construit un graphe **Ndiv-temps** répertoriant les noms désignant des divisions du temps : *an*, *année*, *trimestre*, *mois*, *jour*, *heure*, *minute*, etc.

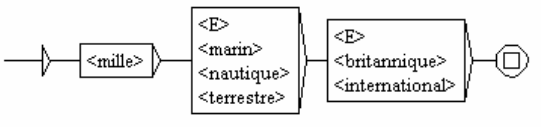

Figure 29 : Mille

Au vu de cette liste, il est clair que nous n'avons pas répertorié toutes les unités simples existantes. Mais, nous considérons que cela est suffisant pour notre étude.

Nous décidons quand même d'ajouter un graphe (**Nmonnaie**) regroupant toutes les unités de monnaie plus le graphe des symboles (**Nmonnaie_abr**). Certains noms de monnaies peuvent être regroupés dans des sous-graphes (**Dinar** pour les différents types de dinars, **Dollar** pour les différents types de dollars, **Franc**, **LivreSterling**, etc.). Nous donnons le graphe **Dollar** dans la figure ci-dessous :

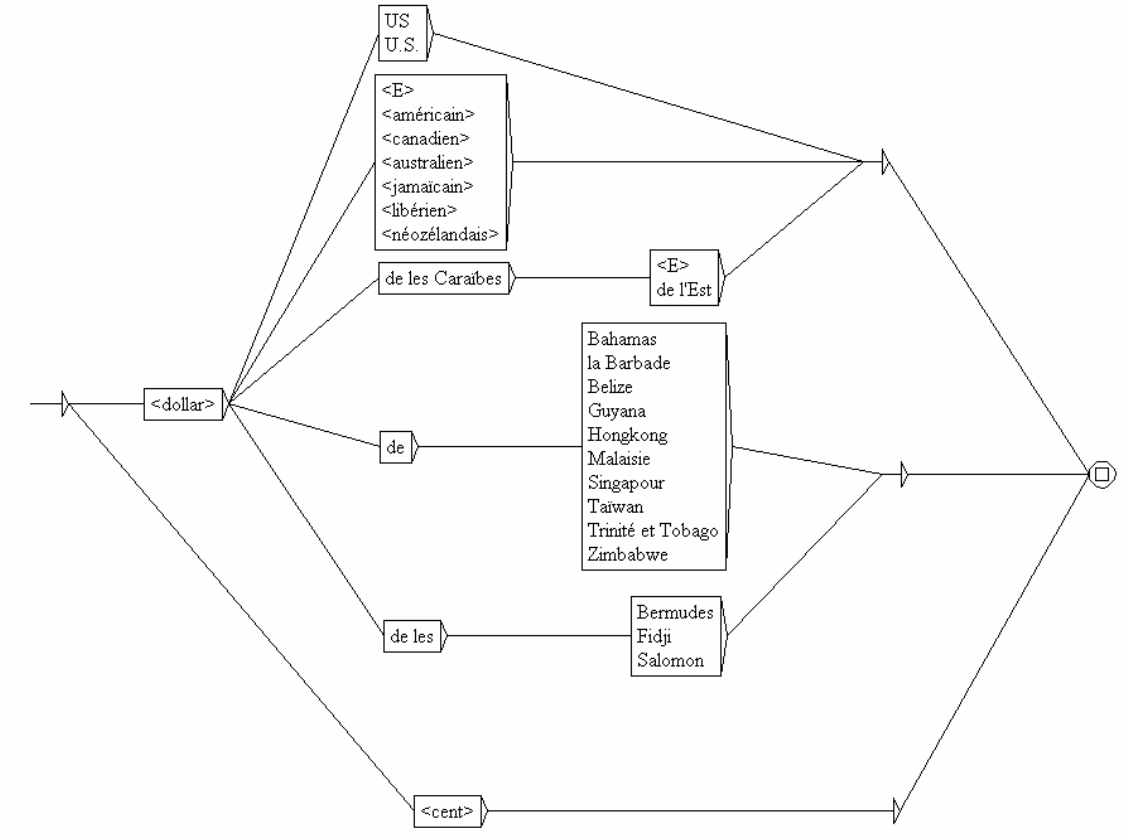

Figure 30 : Dollar

Nous avons également construit le même genre de graphes pour les unités en anglais pour lesquelles nous avons utilisé le dictionnaire en-ligne se trouvant à l'URL www.unc.edu/~rowlett/units/. Nous donnons ci-dessous le graphe décrivant la classe des unités dont l'unité de base est *gram* (gramme). Les symboles de monnaies ont un comportement différent des autres unités car ils sont toujours situés avant *Dnum* (ex : *£10*).

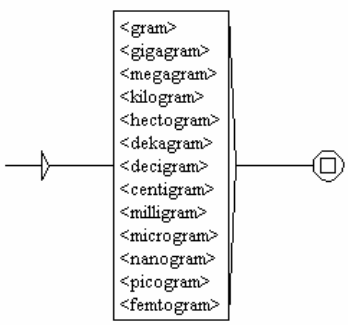

Figure 31 : Gram

3.2.3.2 Les contraintes internes à *Dnum Unité*

La séquence *Dnum Unité* comprenant deux composants indépendants est une facilité théorique d'écriture que l'on s'est donné. Dans les faits, bien que cette indépendance se vérifie souvent, elle n'est pas toujours vraie. En effet, plusieurs points remettent en cause cette représentation. Tout d'abord, il semble exister des contraintes stylistiques entre *Dnum* et *Unité*. Par exemple, la combinaison (*DnumEnLettres + DetNnumDe) Unité_Abr* n'est pas naturelle alors que les autres sont tout à fait acceptables : **(dix + quelques dizaines de) m ; 10 (m + mètres) ; (dix + quelques dizaines de) mètres*. Nous illustrons cette règle sous la forme d'un graphe représentant la séquence formée d'un déterminant numérique et d'une unité métrique.

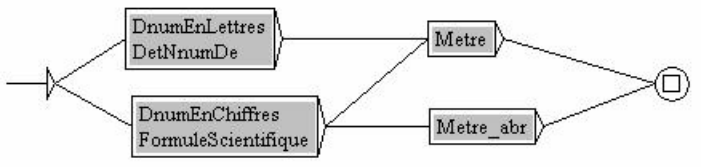

Figure 32 : Règle stylistique

Ensuite, les déterminants numériques ne sont pas toujours connexes et peuvent se diviser en deux parties entre lesquelles vient se greffer l'unité, comme le montrent les exemples ci-dessous :

Max a un retard de huit minutes (trente + et demi) par rapport à son emploi du temps
Marie a sauté 5 m 60

Cette contrainte est représentée par le graphe ci-dessous [35] . On utilise le graphe **NombreEntierEnChiffres** car tout nombre décimal est interdit (** 2,5 m 12*). Notons que l'espace blanc entre l'unité et le nombre entier en chiffres qui la suit est respecté pour éviter la confusion avec les *mètres carrés* (*m2*) ou *mètres cubes* (*m3*) :

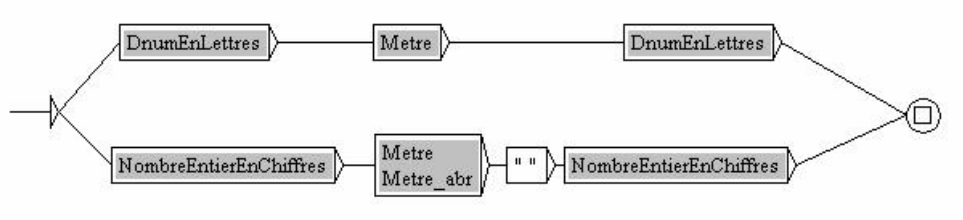

Figure 33 : Non-connexité des déterminants

Enfin, une séquence *Dnum Unité* peut parfois commuter avec une suite de plusieurs *Dnum Unité*. Cette suite a une syntaxe qui lui est propre car elle dépend de la classe d'unités utilisée. Cette forme sert essentiellement à ajouter une précision à la mesure :

Léa a eu droit à trois heures, dix minutes et douze secondes de sueurs froides
*Luc a exactement parcouru 10 kilomètres et (30 mètres + *30 secondes)*

Nous représentons un exemple (très partiel) de telles séquences dans le graphe suivant :

[35] Le graphe présenté est partiel car il ne contient pas compte d'expressions telles que *trois mètres et demi*.

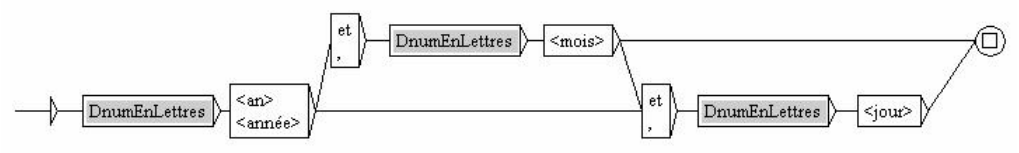

Figure 34 : exemple de syntaxe propre à une unité

Ainsi, pour chaque classe d'unités, il est nécessaire de regrouper la séquence *Dnum Unité* en un seul graphe **GNmesure-unite**, en tenant compte des remarques précédentes et de la présence potentielle de prédéterminants. Nous donnons ci-dessous un exemple d'un tel graphe pour la classe d'unités **Metre**. Le graphe **DnumMetre-precis** reconnaît des suites *Dnum Metre* selon une syntaxe spécifique (ex : *10 kilomètres et 300 mètres*)[36]. Ce dernier graphe a été conçu de telle manière qu'il ne reconnaisse pas des séquences comme *10 kilomètres et 3 000 mètres* (cf. graphe ci-dessous). Les graphes **DnumEnLettres1-99** et **DnumEnLettres1-999** représentent des nombres écrits en toutes lettres respectivement de *un* à *quatre-vingt dix-neuf* et de *un* à *neuf cent quatre-vingt dix-neuf*. Les graphes **NombreEntierEnChiffre1-99** et **NombreEntierEnChiffre1-999** décrivent des nombres entiers écrits en chiffres allant respectivement de *1* à *99* et de *1* à *999* (plus 0).

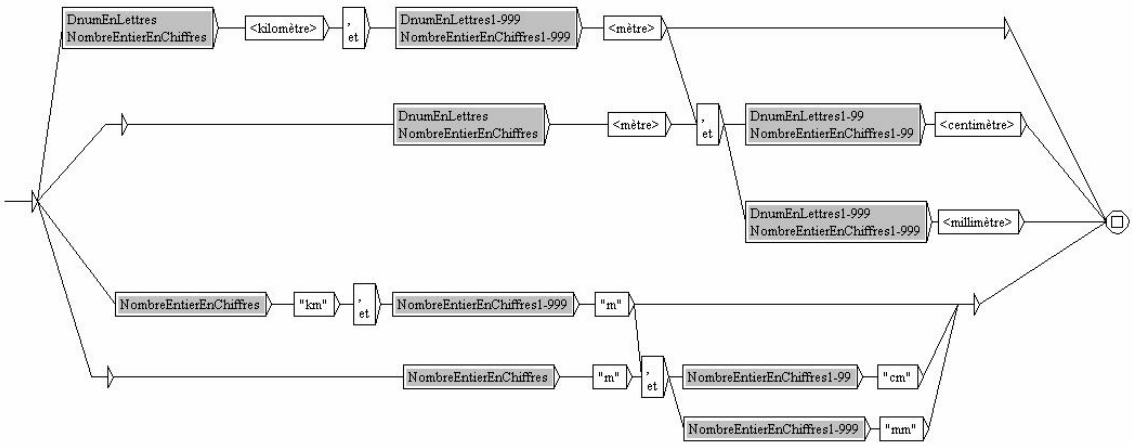

Figure 35 : DnumMetre-precis

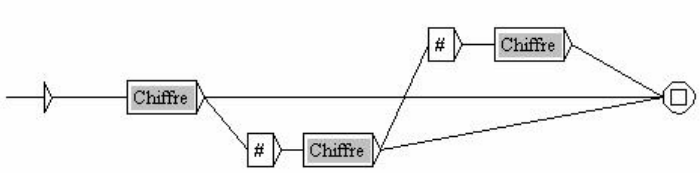

Figure 36 : NombreEnChiffres1-999

[36] D'une manière générale, ce type de graphe est nommé selon le modèle suivant : *DnumUnite-precis*.

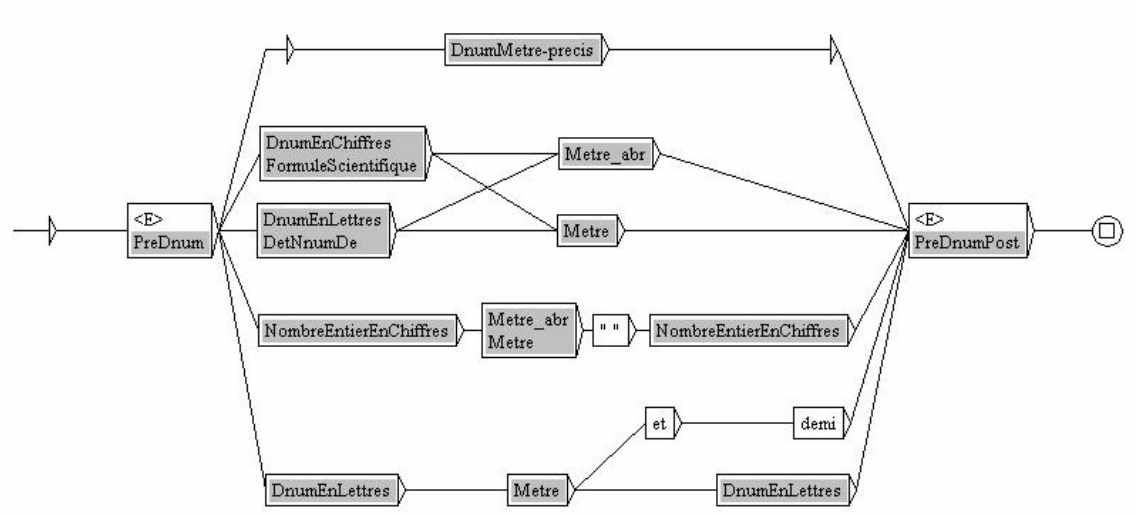

Figure 37 : GNmesure-metre

Dans la suite, nous emploierons le terme générique *GNmesure* pour représenter la séquence *Dnum Unité.*

3.2.3.3 Contraintes entre *Ng* et *Unité*

Dans le schéma de phrase *Det Ng être de Dnum Unité* (=: *la longueur est de 30 m*), chaque *Ng* sélectionne un ensemble restreint d'unités homogènes :

- soit des unités simples comme *largeur* qui sélectionne le *mètre*, ses multiples (*kilomètre*) et ses sous-multiples (*millimètre*), plus d'autres unités comme le *mille nautique* et le *mile*.

- soit des unités complexes (des combinaisons d'unités simples) comme *vitesse* qui sélectionne des combinaisons d'unités de mesure de longueur (*mètre, mile*) et de temps (*heure*) : *kilomètres à l'heure*.

Sur la base de l'étude de J. Giry-Schneider (1991), nous avons systématiquement examiné les noms *Ng* entrant dans nos deux structures de base et nous avons associé à chacun un ensemble de graphes représentant les unités sélectionnées par *Ng*. L'étude nous a conduit à décrire 17 classes d'unités qui sont explicitées dans le tableau ci-dessous. Chaque ligne correspond à une classe. La première colonne donne, pour chaque classe, le nom du graphe de type *GNmesure* qui sera automatiquement construit à partir du contenu de la classe. La deuxième colonne correspond à l'ensemble des noms des graphes[37] décrivant les unités écrites en toutes lettres d'une classe (graphes du type *Unite*). La troisième colonne correspond à l'ensemble des noms des graphes décrivant les symboles des unités d'une classe (graphes du type *Unite_abr*). La dernière colonne correspond à l'ensemble des noms de graphes du type *DnumUnite-precis* associés à une classe d'unités.

[37] Par convention, les noms des graphes sont toujours précédés du symbole '*:*'.

A	B	C	D	E
nom graphe	Unite	Unite_abr	DnumUnite-precis	Exemples
GNmesure-longueur	:Metre+:Mile+:Mille	:Metre_abr+:Mile_abr	:DnumMetre-precis+:DnumMile-precis	*15 mètres*
GNmesure-masse	:Gramme+:Livre+:Tonne	:Gramme_abr+:Livre_abr+:Tonne_abr	:DnumGramme-precis	*15 grammes*
GNmesure-temperature	:DegreCelsius+:Kelvin	:DegreCelsius_abr+:Kelvin_abr	-	*15 °C*
GNmesure-force	:Newton	:Newton_abr	-	*13 kN*
GNmesure-population	:Habitant	-	-	*mille habitants*
GNmesure-energie	:Joule+:Calorie+:ElectronVolt	:Joule_abr+:Calorie_abr+:ElectronVolt_abr	-	*124 kJ*
GNmesure-intensite-elec	:Ampere	:Ampere_abr	-	*0,2 A*
GNmesure-informatique	:Bit+:Octet	:Bit_abr+:Octet_abr	-	*56 ko*
GNmesure-temps	:Seconde+:Ndiv-temps	:Seconde_abr+:Ndiv-temps_abr	:DnumNmesure-temps-precis	*deux minutes*
GNmesure-tension	:Volt	:Volt_abr	-	*110 V*
GNmesure-monetaire	:Nmonnaie	:Nmonnaie_abr	:DnumMonnaie-precis	*cinq dollars*
GNmesure-vitesse	:Nmesure-vitesse	-	-	*120 km/h*
GNmesure-surface	:Nmesure-surface	:Nmesure-surface_abr	:DnumNmesure-surface-precis	*10 hectares*
GNmesure-volume	:Nmesure-volume	:Nmesure-volume_abr	-	*43 m3*
GNmesure-densite-pop	:Nmesure-densite-pop	-	-	*10 habitants au km2*
GNmesure-frequence	:Nmesure-frequence	:Hertz_abr	-	*50 Hz*
GNmesure-angle	:Nmesure-angle	:Nmesure-angle_abr	-	*dix radians*

Table 2 : classes d'unités

La plupart du temps, les graphes sélectionnés correspondent à des unités simples décrites dans la section précédente. D'autres représentent des combinaisons d'unités comme pour le nom *vitesse* qui sélectionne des unités complesxes combinant des unités métriques et de temps. En physique, une unité de vitesse est la « division » d'une unité métrique par une unité de temps : *centimètres par heure*, *km/s*. Dans le langage courant, il existe des variations moins « rigoureuses » telles que *kilomètres à l'heure* ou *kilomètres-heure*. Le premier exemple peut même être réduit à *à l'heure* comme dans l'exemple suivant :

Max roule à une vitesse de 80 kilomètres à l'heure
Max roule à une vitesse de 80 à l'heure

Mais cela n'est valable qu'avec le nom *heure* :

Ce météorite a une vitesse de 1,2 km à la seconde
**Ce météorite a une vitesse de 1,2 à la seconde*

Nous donnons ci-dessous le graphe **Nmesure-vitesse** représentant ce type d'unités. Dans ce graphe, nous autorisons des expressions plus exotiques telles que *miles par an*. Nous ne divisons pas cette unité en deux comme auparavant (unité en toutes lettres ; symboles). Le graphe **Nmesure-longueur** est l'union des graphes **Metre**, **Metre_abr**, **Mile**, **Mile_abr** et **Mille**. Le graphe **Nmesure-temps** est l'union des graphes **Ndiv-temps**, **Ndiv-temps_abr**, **Seconde** et **Secondes**.

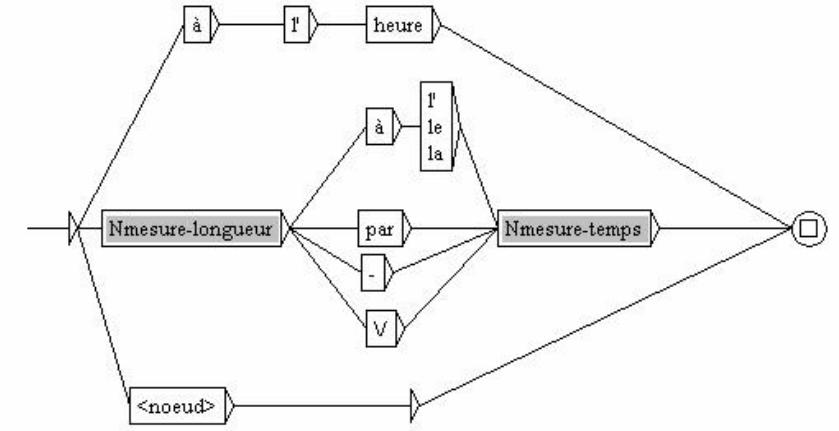

Figure 38 : Nmesure-vitesse

Les noms désignant une surface (*aire*, *surface*, *superficie*) sélectionnent deux types d'unités :

- la combinaison d'une unité métrique de longueur accompagnée soit d'un 2 soudé si l'on a un symbole, soit du modifieur *carré* si l'unité le précédant est écrite en toutes lettres (ex : *m2*, *mètre carré*) ;

- des unités de surface simples comme *are*, *hectare* symbolisées par *a* et *ha*.

Dans ce cas, on sépare les symboles des unités écrites en toutes lettres et on construit les deux graphes **Nmesure-surface** et **Nmesure-surface _abr** suivants :

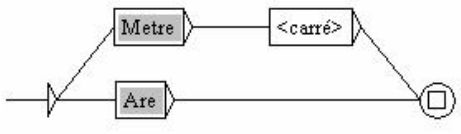

Figure 39 : Nmesure-surface

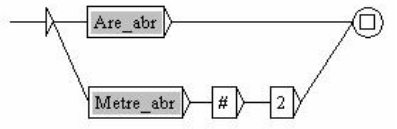

Figure 40 : Nmesure-surface_abr

A cette classe, on associe également le graphe **DnumNmesure-surface-precis**. En effet, on peut exprimer une mesure de surface à l'aide de la multiplication de deux mesures de longueurs (**GNmesure-longueur**) comme suit :

Marie a acheté un champ d'une surface de (70 m x 120 m + cent mètres sur trente).

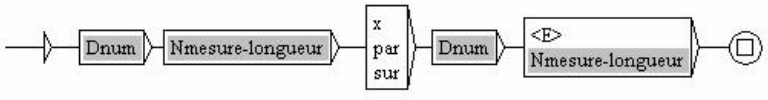

Figure 41 : DnumNmesure-surface-precis

Il en est de même pour les unités sélectionnées par *volume*. Le graphe formé est l'union de deux types d'unités :

- des séquences comprenant une unité métrique de longueur suivie d'un 3 collé ou du modifieur *cube* ;

- Les unités dérivées de *litre* se trouvant dans les graphes **Litre** et **Litre_abr**.

Nous synthétisons ces unités dans les graphes **Nmesure-volume** et **Nmesure-volume_abr**.

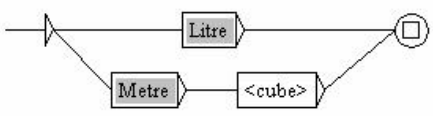

Figure 42 : Nmesure-volume

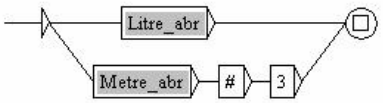

Figure 43 : Nmesure-volume_abr

Ci-dessous nous donnons le graphe **Nmesure-densite-pop** décrivant les unités sélectionnées par le nom *densité (démographique + de population)* et reconnaissant des expressions telles que *habitants au km2*.

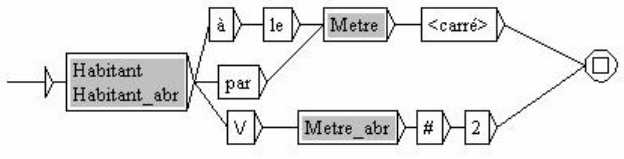

Figure 44 : Nmesure-densite-pop

L'unité scientifique traditionnellement associée à la fréquence est le *hertz*. Ainsi, *fréquence* sélectionne les unités décrites par le graphe **Hertz**. Cependant, dans le langage courant, c'est beaucoup plus libre et cela peut être n'importe quel groupe nominal comptable (sans déterminant) suivi par une préposition (ou le symbole '/'), un déterminant optionnel et une unité de temps :

Le moteur a une fréquence de trois tours par seconde
La machine a une fréquence de trente poulets (à la + /) minute

Ainsi, nous avons besoin d'une description complète d'un groupe nominal. Cependant, pour des raisons évidentes de clarté ce groupe nominal ne peut être trop long car il doit être suivi d'une séquence exprimant le temps. Ainsi, nous limitons notre groupe nominal sans déterminant à la séquence maximale suivante *Adj N Adj de Det N*. Les unités associées à *fréquence* sont représentées dans le graphe **Nmesure-frequence** où les symboles <*A*>, <*N*> et <*DET*> désignent respectivement un adjectif, un nom et un déterminant.

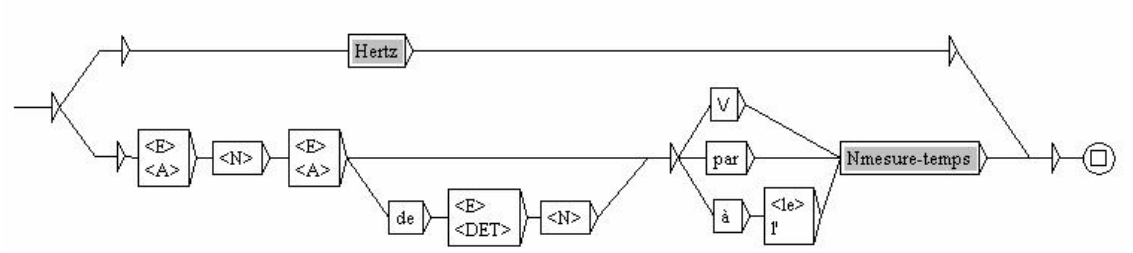

Figure 45 : Nmesure-frequence

Le graphe **Nmesure-angle** contient les unités *radian, degré* et leurs sous-multiples comme *minute*. Les symboles sont dans **Nmesure-angle_abr** : °, *rad*, etc.

Certains *Ng* sont ambigus comme *tension* qui désigne soit une *tension électrique* soit une *tension artérielle*. Cette ambiguïté est levée si l'on regarde les unités sélectionnées par

chacun : *Volt* pour *tension électrique* et *<E>* pour *tension artérielle*. Par ailleurs, *longueur* est aussi ambigu : il peut désigner

- soit une durée comme dans :

  Ce spectacle a une longueur de 2 heures

- soit une mesure métrique comme dans :

  Cette corde a une longueur de 25 mètres

Il en est de même pour *poids* qui est soit une force (nom standard en physique), soit une masse comme il est courant de l'utiliser dans la vie de tous les jours. La vie courante ne permet pas de distinguer les notions de force et de masse.

Les noms sélectionnant une unité monétaire ont un comportement particulier par rapport aux autres en ce sens qu'ils autorisent l'effacement de l'unité :

Ce cadeau a une valeur de 2,50 (euros + ?E)

3.2.3.4 Processus de génération des graphes *GNmesure*

La table décrivant les classes d'unités peut être vue comme une table syntaxique (cf. chapitre sur le lexique-grammaire). Chaque ligne correspond à une entrée lexicale (un nom de classe) qui est donnée dans la première colonne (ex : *GNmesure-longueur* pour la première ligne). Les colonnes (autres que la première) contiennent certaines propriétés de la classe : colonne B, l'ensemble des graphes d'unités en toutes lettres de la classe ; colonne C, l'ensemble des graphes des symboles (s'ils existent) ; colonne D, l'ensemble des graphes du type *DnumUnite-precis* (s'ils existent). Chaque élément de la table est soit un élément lexical (noms des graphes) soit un booléen (+ pour vrai ; - pour faux)[38].
Pour chaque entrée lexicale de la table, le but est de construire un graphe qui décrit tous les groupes nominaux de mesure de type *GNmesure* dans lesquelles rentrent cette entrée. Nous utilisons la méthode d'E. Roche (1993, 1994) qui consiste à utiliser un graphe patron qui représente l'ensemble des structures potentielles des entrées de la table. Un graphe patron est associé à une classe d'éléments lexicaux ; il est paramétré de façon à pouvoir être adapté à chaque élément lexical en fixant la valeur des paramètres. Chaque élément d'information des tables (c'est à dire une propriété ou plus simplement une colonne de la table) est représenté par une variable dans le graphe patron. Pour chaque ligne *i* de la table à convertir *T*, on réalise une copie du graphe patron. Puis, pour chaque transition (ou boîte) de ce graphe, nous effectuons les opérations suivantes pour chacune des variables *@j* contenues dans cette transition :

- si $T(i,j) = +$, on remplace *@j* par le mot vide <E> ;
- si $T(i,j) = -$, on supprime la transition, coupant ainsi le chemin reconnaissant la structure correspondant à la colonne *j* ;
- sinon, on remplace *@j* par l'information lexicale contenue dans $T(i,j)$.

Par cette méthode, nous générons les graphes de type *GNmesure* associés à notre table. Nous utilisons le graphe patron ci-dessous. La variable *@B* correspond aux informations contenues

[38] Cette table ne contient pas de + ; les – signifient qu'il n'existe pas de graphes du type défini par la colonne.

dans la colonne B, la variable *@C* correspond aux informations contenues dans la colonne C, etc. Pour chaque entrée le graphe généré a pour nom le contenu de la colonne A (*@A*).

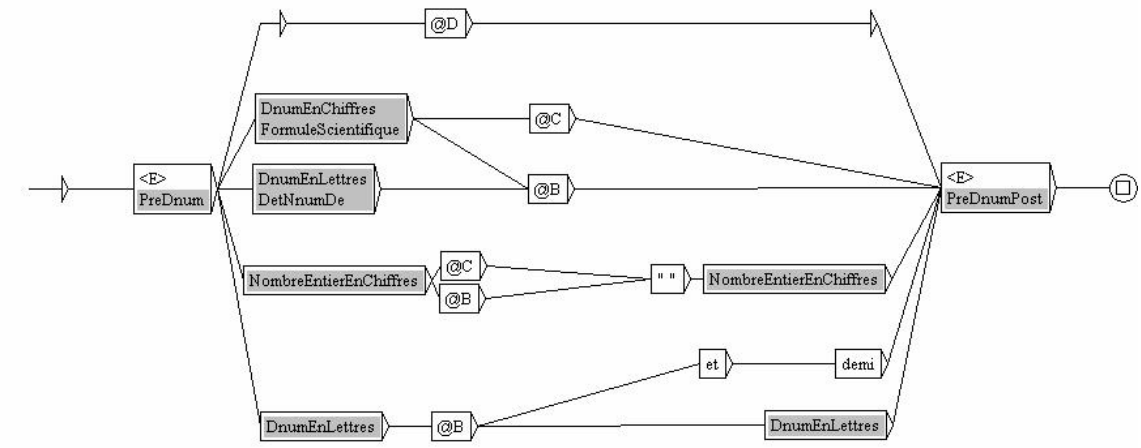

Figure 46 : Graphe patron pour générer les graphes du type *GNmesure*

Pour l'entrée *GNmesure-vitesse*, nous obtenons le graphe **GNmesure-vitesse** ci-dessous. La variable *@B* est remplacée par l'information lexicale : *Nmesure-vitesse* (nom du graphe précédé de *:*). Les boîtes contenant les variables *@C* et *@D* sont supprimées, éliminant ainsi les chemins reconnaissant des structures interdites.

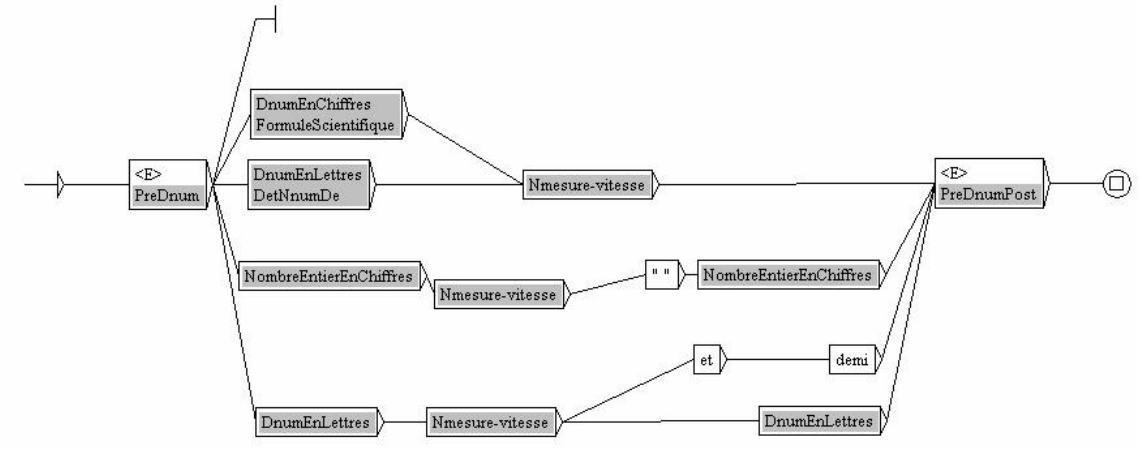

Figure 47 : GNmesure-vitesse

3.2.4 Quelques variantes

3.2.4.1 Quelques variantes simples

On constate que nos phrases peuvent être étendues à d'autres schémas de phrase :

(a) *N*0 *être Adj à N*1 *= : la longueur est supérieure à 10 mètres*
(b) *N*0 *être Prép N*1 *= : le poids est de l'ordre de 10 kilos*
(c) *N*0 *être Vpp à N*1 *= : la tension est limitée à trente volts*
(d) *N*0 *V Prép N*1 *= : la fréquence atteint 55 Hz*

La structure (a) contient des adjectifs appropriés aux mesures tels que *égal* et *supérieur*. Ils sont décrits dans le graphe **Adj-numA**.

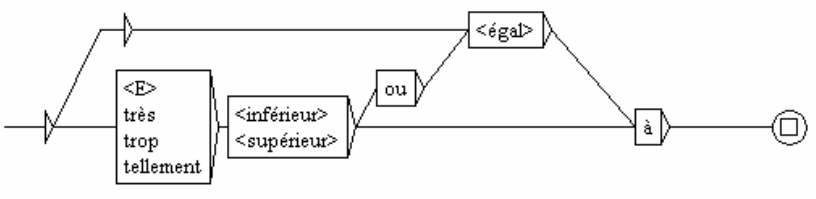

Figure 48 : Adj-numA

La structure (b) comprend des prépositions composées (pour la plupart) qui peuvent aussi être vues comme des prédéterminants (ex : *jusqu'à* , cf. M. Gross, 1977). Ces prépositions sont décrites dans le graphe **PreDnumPrep**.

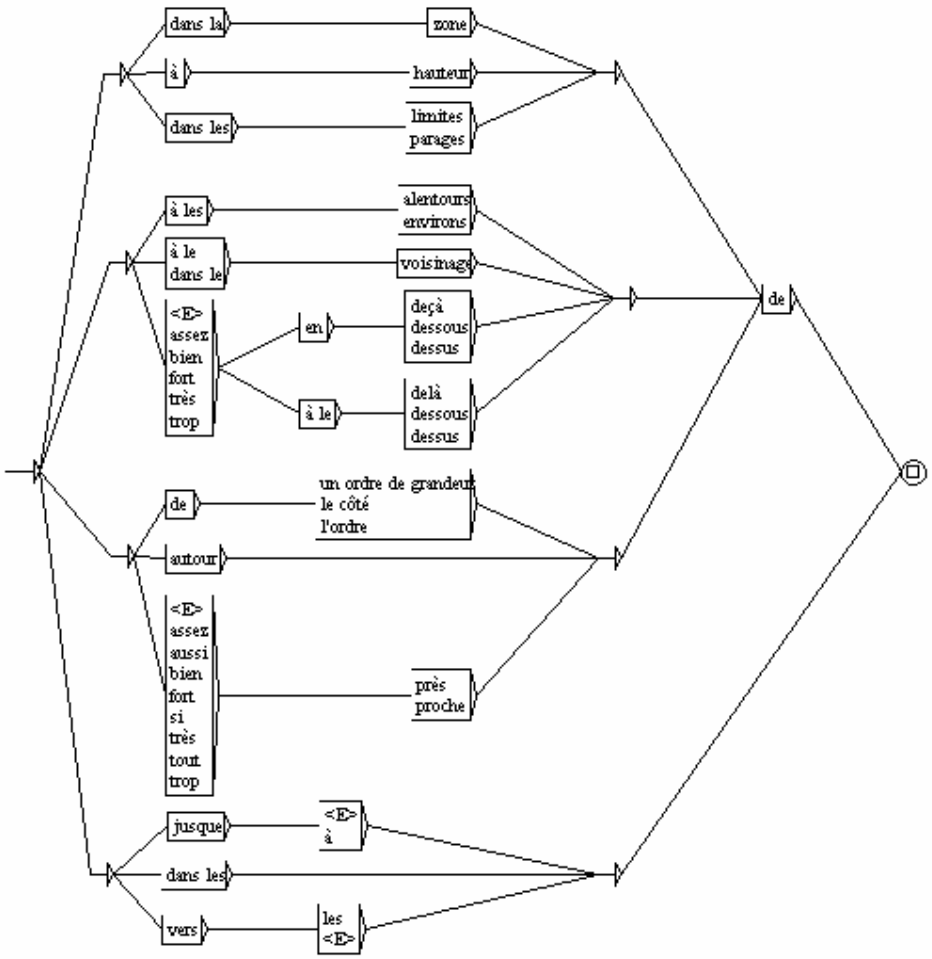

Figure 49 : PreDnumPrep

La structure (c) comporte des verbes au participe passé *Vpp* : *la tension est limitée à trente volts*. Cette dernière phrase est la forme passive de

On limite la tension à trente volts

Ces passifs ont un sens statif, même avec agent. Nous décrivons quelques *Vpp* de ce type dans le graphe **Vpp-numA**.

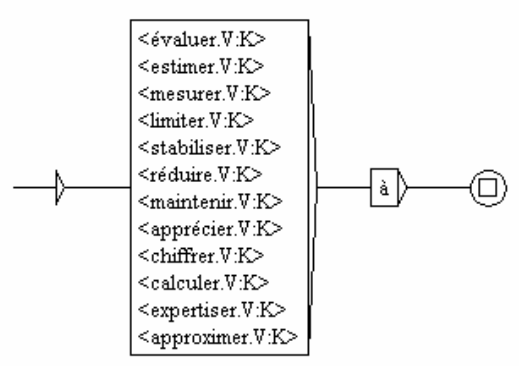

Figure 50 : Vpp-numA

Enfin, il existe quelques verbes appropriés aux mesures. Ils ont tous un aspect statique avec quelques nuances sémantiques : *atteindre, s'élever à, se monter à*.

3.2.4.2 Quelques variantes complexes

Dans cette section, nous nous intéressons à des combinaisons plus complexes que dans les sections précédentes. Nous regardons d'abord le comportement de nos phrases de mesure lorsqu'on leur applique des coordinations. Nous examinons aussi en détail les structures *de … à …* et *entre … et …* qui, dans notre cas, désignent des approximations de mesures sous la forme d'intervalles. Nous montrons qu'elles ne peuvent être décrites localement de manière complète : le contexte minimal est une phrase.

3.2.4.2.1 Les coordinations

Prenons les deux phrases simples de mesure suivantes :

Max a une taille de 1,70 m
Luc a une taille de 1,80 m

Ces phrases peuvent être coordonnées à l'aide de la conjonction *et* :

Max a une taille de 1,70 m et Luc a une taille de 1,80 m

Nous réduisons cette phrase complexe à une forme factorisée, en plusieurs étapes, à l'aide de l'adverbe *respectivement* :

- regroupement des sujets :

  Max et Luc ont respectivement une taille de 1,70 m et une taille de 1,80 m

- factorisation de *taille* (*N*) et de la préposition *de* :

  Max et Luc ont respectivement une taille de 1,70 m et (de + E) 1,80 m

- factorisation de l'unité lorsqu'elle est similaire

  Max et Luc ont respectivement une taille de 1,70 et 1,80 m

Cette dernière étape n'est pas réalisable lorsque les unités ne sont pas les mêmes :

La première et la deuxième ont respectivement une tension de 30mV et 3V

Cette analyse est aussi valable pour les noms entrant dans la deuxième structure étudiée. Nous avons les deux cas suivants

Pierre est à une distance de 30 km de Paris et Pierre est à une distance de 50 km de ta ville
= Pierre est à une distance de 30 et 50 km respectivement de Paris et de ta ville

Pierre est à une distance de 200 km de Paris et Paul est à une distance de 45 km de Paris
= Pierre et Paul sont respectivement à une distance de 200 et 45 km de Paris

La conjonction *ou* marque une approximation de la valeur numérique du déterminant (sous la forme d'un choix entre plusieurs valeurs) :

Max a cinq ou six ans.

Nous pouvons interpréter *cinq ou six* comme un déterminant composé, mais cela peut poser des problèmes. En effet, la phrase ci-dessus est équivalente aux deux phrases suivantes :

Max a cinq ans ou six ans.
Max a cinq ans ou Max a six ans.

On peut aussi avoir des phrases du type :

Le tuyau est à une distance de 90 cm ou 1 m du mur.
Le tuyau est à une distance de 90 cm du mur ou le tuyau est à une distance de 1 m du mur.

Pour plus de détails sur la coordination dans les groupes nominaux, nous suggérons au lecteur de se référer à C. Domingues (2001).

3.2.4.2.2 La structure entre … et …

Revenons à notre phrase de base *Det Ng être de Dnum Unité* :

Cette longueur est de 15 m.

Il est possible d'exprimer une approximation en utilisant un intervalle de mesures grâce à la structure *être compris entre … et ….* On remarque que *compris* peut être effacé.

Ce Ng être (E + compris) entre $Dnum_1$ $Unité_1$ et $Dnum_2$ $Unité_2$
= : cette longueur est (E + comprise) entre 90 cm et 1,10m

Comme pour les coordinations, $Unité_1$ et $Unité_2$ peuvent être factorisées si $Unité_1 = Unité_2$:

Det Ng être (compris + E) entre $Dnum_1$ (E + Unité) et $Dnum_2$ Unité
= : la longueur est (comprise + E) entre 90 (E + m) et 110 m

La réduction à un groupe nominal par relativation puis réduction de la relative donne des séquences telles que :

Une longueur (E + comprise) entre 90 (E + m) et 110 m

Lorsque l'on utilise une variante de *être de*, comme *être supérieur à* (cf. section précédente), on observe un comportement un peu différent à cause de la présence de la préposition *à* : la préposition *à* est obligatoirement effacée. Les phrases obtenues ne sont pas très naturelles.

La température est supérieure à dix degrés Celsius.
? La température est supérieure (E + à) entre dix et quinze degrés Celsius*

Par ailleurs, la variante en *être PreDnumPrep* est peu naturelle :

La tension est (de l'ordre de + dans les + à hauteur de) 15 V
*?*La tension est (de l'ordre d' + dans les + à hauteur d') entre 14 et 15 V*

Pour les variantes avec des verbes tels que *atteindre*, on a :

*La diamètre atteint (*comprise + E) entre 90 (E + m) et 110 m*

Les structures *entre Dnum et Dnum Unité* sont beaucoup plus fréquentes que *entre Dnum Unité et Dnum Unité*. Ne tenir compte que des expressions trouvées dans les corpus aussi grands soient-ils n'est pas suffisant. La première intuition à partir des expressions du corpus est de considérer *entre Dnum et Dnum* comme l'équivalent d'un déterminant numérique comme montré dans l'analyse ci-dessous :

(Le diamètre) (atteint) ((entre 90 et 110) m)

Nos graphes permettent d'analyser des formes plus rares telles que *entre 90 cm et 1,10m*. L'exemple ci-dessus est analysé comme suit :

(Le diamètre) (atteint) (entre (90) et (110 m))

3.2.4.2.3 La structure de … à …

Il existe une autre structure permettant d'exprimer une approximation de mesure sous la forme d'un intervalle : c'est la séquence *de … à …*. Nous modifions notre phrase de départ en la remplaçant par :

Det Ng être de $Dnum_1$ $Unité_1$ à $Dnum_2$ $Unité_2$
=: ? cette température est de 10 à 15 degrés

Notre exemple ci-dessus est réductible au groupe nominal :

cette température (qui est + E) de 10 degrés à 15 degrés

Comme pour *entre… et …*, lorsque l'on utilise les variantes *être Adj à*, la préposition *à* est interdite. Néanmoins, l'exemple ci-dessous montre que l'utilisation de telles phrases n'est pas très naturelle :

*La longueur est supérieure (?E + *à) de 10 m à 15 m*

Avec les *PreDnumPrep*, on observe l'effacement obligatoire de la préposition *de* :

*La longueur est (à hauteur de + de l'ordre de + ?vers les) (E + *de) 10 à 15 m*

Avec les verbes sans préposition, la préposition *de* n'est pas obligatoire :

La température atteint (de + E) 10 degrés à 15 degrés

La structure d*e … à …* est naturellement ambiguë. Son interprétation dépend du verbe de la phrase élémentaire dans laquelle elle se trouve. En effet, elle peut exprimer une évolution et non une approximation sous la forme d'un intervalle comme avec le verbe *passer* :

Le prix du pain est passé de 65 à 70 centimes

Dans cette phrase, le prix initial du pain est de 65 centimes ; à l'état final, il est de 70 centimes. Dans beaucoup de cas, cette ambiguïté est localement impossible à lever comme avec le verbe *augmenter* :

La tension entre ces deux points de la ligne a augmenté de 10 (E + V) à 15 V

Il existe deux analyses :

La tension entre ces deux points de la ligne a augmenté d'une valeur de 10 (E + V) à une valeur de 15 V
La tension entre ces deux points de la ligne a augmenté d'une valeur de 10(E + V) à 15 V

3.2.4.2.4 Le tiret

Il existe par ailleurs d'autres structures combinant des nombres exacts et désignant une approximation de valeur sous la forme d'un intervalle. La plus simple est l'emploi du tiret entre deux nombres :

L'intensité du courant sur cette ligne est de 150-200 ampères

Ces phrases paraissent plutôt orales qu'écrites. Il peut exister des problèmes d'interprétation car le nom-unité *ampères* est effacé entre 150 et le tiret :

L'intensité du courant sur cette ligne est de 150 ampères-200 ampères

Cela est confirmé par la phrase :

Ce chemin fait 800 mètres-1 kilomètre.

3.2.4.2.5 *Remarques*

- Nouvelle notation

Dorénavant, dans les graphes, pour décrire la séquence *Dnum Unité*, nous employons les termes *GNmesure* et *GNmesureFinal* qui sont aussi les noms génériques des graphes que nous utilisons. Dans *GNmesure*, l'unité est optionnelle alors qu'elle est obligatoire dans *GNmesureFinal* (ex : *entre GNmesure et GNmesureFinal*).

- Problèmes stylistiques de la séquence *Dnum Unité*

Comme nous l'avons mentionné, un nombre en lettres ne peut pas être suivi d'une unité sous la forme d'un symbole alors qu'un nombre en chiffres peut être suivi par n'importe quel type d'unités : *deux mètres* ; **deux m* ; *2 m* ; *2 mètres*. Par ailleurs, les combinaisons complexes requièrent une certaine homogénéité dans le choix des types de déterminants numériques et d'unités. Par exemple, il semble difficilement concevable d'avoir la séquence suivante : *entre 4 et cinq mètres*. Nos graphes ne tiennent pas compte de cette dernière règle par souci de clarté et de simplicité. Ce choix peut causer quelques rares erreurs de reconnaissance dans les textes.

- Ambiguïté des combinaisons complexes

Il y a une ambiguïté dont nous n'avons pas tenu compte dans nos graphes. En effet, la séquence *entre 4 et 5 millions de dollars* est interprétée comme *entre 4 dollars et 5 millions de dollars*. Mais il existe une autre interprétation qui est, dans la quasi-totalité des cas, la bonne (à cause du point précédent : homogénéité des déterminants numériques). Il faut considérer que c'est la séquence *millions de dollars* qui a été factorisée et non *dollars*. La séquence précédente doit alors être analysée comme *entre 4 millions de dollars et 5 millions de dollars*.

- Récursivité dans les combinaisons complexes

Les combinaisons complexes en *entre … et …* et *de … à …* sont théoriquement récursives. En effet, la règle récursive suivante semble pouvoir s'appliquer : *Dnum* → *entre Dnum et Dnum*. Cependant, le nombre de niveaux est très étroitement limité : l'effacement du *N* =: *valeur* est interdit.

> *?* La tension est entre [de (10 V) à 31 V] et [de (4 kV) à (5 kV)]*
> *La tension est entre une valeur de 10 à 30 V et une valeur de 4 à 5 kV*

Notre choix de ne pas décrire récursivement ce type de séquences paraît donc fondé. Notre représentation est donc équivalente à un automate fini.

3.3 Représentation des mesures absolues

3.3.1 Généralités

La structure élémentaire qui nous intéresse représente l'expression de la mesure (absolue) d'une caractéristique ou propriété intrinsèque (désignée par *Ng*) d'un élément *N0* :

*N*0 *avoir un Ng de Dnum Unité*
Le bateau a une longueur de 15 mètres

La première phrase indique que *le bateau a une longueur* (i.e. *N*0 a une caractéristique *Ng*) et puis que *cette longueur est de 15 mètres* (i.e. mesure de la caractéristique *Ng*).
Nous avons étendu les résultats de J. Giry-Schneider (1991) à un plus grand nombre de noms et de propriétés syntaxiques associées. Nous regardons aussi l'application de cette analyse à la reconnaissance automatique de ce type d'expressions dans des textes.

3.3.2 Ng composés

Dans un premier temps, nous avons sélectionné un ensemble de caractéristiques *Ng* qui entrent dans cette structure de phrase. Nous prenons les plus courantes : longueur, poids, coût, température, etc. Au total, nous en avons sélectionné une quarantaine. Nous regardons également quelques noms du domaine économique comme *loyer* pour l'exemple, mais ceci aurait nécessité une étude bien plus approfondie. Nous renvoyons aux travaux de M. Gross (1997) et de T. Nakamura (à paraître) sur le domaine de la bourse. Les noms simples sont, par exemple, les noms *longueur*, *poids*, *vitesse*, *force*, *coût*, etc. Plus des deux-tiers ont cette forme. D'autres sont des mots composés. Nous pouvons les classer en plusieurs classes selon leur structure interne (G. Gross, 1996) :

- une classe NA (nom suivi d'un adjectif) : *tension électrique* ; *tension artérielle* ; *densité démographique* ; *pression atmosphérique* ; *intensité lumineuse* ; *intensité électrique* ; *puissance énergétique* ; *loyer mensuel*

- une classe NDN (nom suivi de la préposition *de* puis d'un nom) : *taille de chaussure* ; *pointure de pied(s)* ;

- une classe complexe NDNA : *taille de mémoire (vive + cache + virtuelle)*

La quasi-totalité de ces noms composés se retrouvent dans les textes sous une forme simple résultant d'un effacement d'un ou plusieurs composants du mot composé. En général, c'est la partie à droite du premier nom qui est effacée comme par exemple dans :

Max a une tension (artérielle + E) de 10
Paris a une densité (démographique + E) de 10 000 hab/km2
Luc a une pointure (de pieds + E) de 43

Certains noms composés du domaine de l'informatique (ex : *taille de mémoire vive*) ont des formes ambiguës particulières. La préposition *de* peut être effacée si l'on supprime l'adjectif :

Cette machine a une taille de mémoire vive de 128 Mo
*Cette machine a une taille mémoire (E + *vive) de 128 Mo*

Le nom de tête *taille* peut aussi disparaître dans la séquence d'origine ; l'effacement de l'adjectif y est toujours possible :

Cette machine a une mémoire (E + vive) de 128 Mo

Notons pour finir que *loyer mensuel* se comporte différemment et ne peut être traité qu'au niveau de la phrase (et non localement comme pour les précédents). En effet, l'adjectif *mensuel* peut à la fois être effacé et transformé en adverbe (*mensuellement*) pouvant s'insérer n'importe où dans la phrase.

Marie a un loyer (mensuel + E) de 1 000 euros
Mensuellement, Marie a un loyer de 1 000 euros

Il en est de même pour d'autres noms dénotant des flux journaliers, mensuels, annuels, etc. : *débit, flux*, … Ce comportement est impossible pour les autres noms composés de la classe NA :

Max a une tension (artérielle + E) de 10
** Artériellement, Max a une tension de 10*

Paris a une densité (démographique + E) de 10 000 hab/km2
? Démographiquement, Paris a une densité de 10 000 hab/km2*

Par la suite, nous décidons de ne pas traiter les noms tels que *loyer*.

3.3.3 Propriétés distributionnelles, lexicales et transformationnelles

Nous étudions maintenant les variations lexicales et les transformations que peut subir notre phrase de base :

N0 avoir un Ng de Dnum Unité

3.3.3.1 Distribution du sujet

La distribution du sujet dépend du nom. Nous distinguons trois types de sujets : les groupes nominaux humains (*Nhum*), les groupes nominaux concrets (*Nconc*) et les groupes nominaux prédicatifs (*Npred*). Par exemple, le nom *durée* ne sélectionne ni les sujets humains et ni les sujets concrets :

*(Le spectacle + *Paul + *La corde) a une durée de dix minutes*

Le nom composé *tension artérielle* ne sélectionne que les noms humains alors que *taille* interdit les noms prédicatifs :

*(*Le spectacle + Paul + *la corde) a une tension de 12*
*(*Le spectacle + Paul + la corde) a une taille de 2 m*

La distribution du sujet permet de lever l'ambiguïté du nom *longueur*. En effet, l'une des deux entrées a la même distribution du sujet que *durée*, alors que l'autre a la même distribution que *taille*.

*(Le spectacle + *Paul + *la corde) a une longueur de dix minutes*

*(*Le spectacle + La baleine + la corde) a une longueur de 20 m*

3.3.3.2 Verbes supports

D'abord, le schéma de phrase précédent est un cas particulier du schéma de phrase ci-dessous :

*N*0 *Vsup Prép un Ng de Dnum Unité*

Le verbe support et la préposition associée peuvent connaître des variations lexicales (*avoir, faire, être (à + de), compter, contenir*, etc.) :

Cet immeuble a une hauteur de 150 mètres
= Cet immeuble fait une hauteur de 150 mètres (*Vsup =*: *faire*)

La salle des fêtes a une température de 30°C
= la salle des fêtes est à une température de 30°C (*Vsup =*: *être à*)

Ce spectacle a une longueur de deux heures
= Ce spectacle est d'une longueur de deux heures (*Vsup =*: *être de*)

L'agglomération parisienne a une population de dix millions d'habitants
= L'agglomération parisienne compte une population de dix millions d'habitants (Vsup =: *compter*)

Ces aliments ont une énergie de 10 kJ
= Ces aliments contiennent une énergie de 10 kJ (*Vsup = : contenir*)

Cependant, cette variation dépend du *Ng*. Une étude systématique est nécessaire :

*Max (a + fait + est de + *est à[39] + *comporte) une taille de 2 m*
*Cette propriété (a + fait + ?est de + *est à + ?comporte) une surface de 2 hectares*
*Cette salle (a + fait + ?*est de + est à + *comporte) une température de 17°C*
*Ces aliments (ont + *font + *sont de + *sont à + comportent) une énergie de 10 kJ*
*Ce bus (a + *fait + *est de + est à + *comporte) une vitesse de 100 km/h*

Notons que l'utilisation de *température* avec le verbe support *faire* est autorisée dans une phrase au sujet impersonnel de la forme :

*Il faire Dnum Unité Loc N*0
= : Il fait 10°C dans cette salle

Ce phénomène semble marcher pour *pression* et *hygrométrie* mais il n'existe pas pour les autres noms :

** Il fait une longueur de 100 m (sur + dans) le bateau*
** Il fait une tension de 50 kV (sur + dans) cette ligne*

[39] Il existe un emploi de *être à* qui fonctionne dans ce cas mais il dénote un état temporaire dans une évolution : *Dans sa phase de croissance, Max est (déjà + E) à une taille de 1,70 m.*

3.3.3.3 Permutations

Nous regardons maintenant les transformations que peuvent subir les phrases dont la structure est :

N0 Vsup Prép un Ng de Dnum Unité

Tout d'abord, les séquences *Dnum Unité* et *Ng* peuvent être permutées, le déterminant *un* étant effacé. Cette permutation ne fonctionne pas pour tous les *Ng* :

L'immeuble fait une hauteur de 100 m
= l'immeuble fait 100 m de hauteur

*Le courant (a + *fait) une fréquence de 500 Hz*
**Le courant (a + fait) 500 Hz de fréquence*

Cette propriété est un autre moyen de distinguer les deux emplois de *tension* car ils n'ont pas le même comportement :

Max a une tension de 12
= Max a 12 de tension

La ligne (fait + a) une tension de 220V
*= ?*La ligne (fait + a) 220 V de tension*

Notons que l'utilisation de certains *Vsup* est plus naturelle que pour d'autres ; c'est le cas de *faire* :

Le bateau (a + fait) une longueur de 110 m
Le bateau (?a + fait) 100 m de longueur

Ces permutations sont parfois accompagnées d'accidents morphologiques. Certains *Ng* sont parfois remplacés par des adjectifs morphologiquement associés (*Ng-a*). Ce n'est d'ailleurs le cas que pour les *Ng* sélectionnant une unité métrique :

La corde fait 10 m de long
Le gratte-ciel fait 200 m de haut
La piscine fait 20 m de large

Ce phénomène ne fonctionne pas pour *épaisseur* :

Le mur a une épaisseur de 30 cm
Le mur a 30 cm d'épaisseur
**Le mur a 30 cm d'épais*[40]

Le nom *profondeur* (*Ng-a* =: *profond*) subit un accident plus étrange encore car il peut être remplacé par le nom *fond* (noté *Ng'*) :

Le bassin a 3 m de profondeur

[40] Cette dernière phrase est acceptée en français du Québec.

**Le bassin a 3 m de profond*
Le bassin a 3 m de fond

Notons que le nom *fond* ne rentre pas dans la phrase de base équivalente :

? Le bassin a un fond de 3 m*

Cette propriété permet également de distinguer les deux emplois de *longueur* car ils n'ont pas le même comportement :

Le bateau fait 100 m de long
** Le spectacle fait 2 heures de long*

Notons qu'il est possible de substituer la séquence *en Ng* à la séquence *de Ng*, lorsque l'on a le verbe support *faire*. Cette séquence a alors le comportement d'un adverbe car elle peut s'insérer n'importe où dans la phrase :

La piscine fait 50 mètres (de + en) longueur
*(En + ?*de) longueur, la piscine fait 50 mètres*

3.3.3.4 Nominalisation et adjectivation

Notre phrase de base est également sujette à une adjectivation :

*N*0 *Vsup (Prep) un Ng de Dnum Unité*
*= N*0 *être Ng-a de Dnum Unité*[41]

La cour est large de 50 m
? Le bassin est volumineux de 40 litres*

Tous les *Ng* ne possèdent pas de *Ng-a* associé : *tension*, *température*, etc.

Notre phrase de base peut aussi être transformée en une phrase à prédicat verbal *Ng-v* où *Ng-v* est le verbe morphologiquement associé à *Ng* :

*N*0 *Vsup (Prep) un Ng de Dnum Unité*
*= N*0 *Ng-v Dnum Unité*

Max a un poids de 30 kg
= Max pèse 30 kg

la chaise a un coût de trente euros
= la chaise coûte trente euros

Le nom *population* a un comportement différent car le sujet de *Ng-v* est *Dnum Unité* et son objet est *N*0 :

*N*0 *Vsup (Prep) un Ng de Dnum Unité*
*= Dnum Unité Ng-v N*0

[41] *Ng- a* est l'adjectif morphologiquement lié à *N*.

Le village a une population de 300 habitants
= 300 habitants peuplent le village

Nous remarquons après une analyse quasi-exhaustive que l'intersection entre l'ensemble de nos *Ng* qui entrent dans une structure adjectivale et l'ensemble de nos *Ng* qui entrent dans une structure verbale est vide.

3.3.3.5 Effacement du nom prédicatif

Dans notre structure de base, le nom *Ng* peut être effacé, mais cela dépend d'abord du *Ng* et du *Vsup* :

N0 Vsup (Prep) un Ng de Dnum Unité
= N0 Vsup (Prep) Dnum Unité

Cette corde fait une longueur de trente mètres
Cette corde fait trente mètres

Cette ligne (a + ?fait) une tension de 220V
Cette ligne fait 220V

Cette transformation est difficilement réalisable avec le verbe support *avoir*, excepté pour quelques noms comme *âge*.

Cette corde a une longueur de trente mètres
**Cette corde a trente mètres*

Max a un âge de 10 ans
Max a 10 ans

Le verbe support *faire* est souvent le plus naturel, mais le verbe *être à* est aussi possible :

La salle de classe (fait + est à) une température de 20°C
La salle de classe est à 20°C
la salle de classe fait 20°C

Certains *Ng* ne s'effacent pas (ou très difficilement) comme *périmètre*.

La piscine a un périmètre de 30 m
** La piscine fait 30 m*

On s'aperçoit que la propriété dépend aussi de la nature de *N*0. On ne comprend la phrase *la corde fait 30 m* que parce qu'une corde a la propriété d'être longiligne et on en déduit que la caractéristique mesurée est la longueur. Pour la vitesse, on observe un phénomène particulier : l'effacement de vitesse dans la phrase de base est interdit sauf dans le cas exceptionnel où *N*0 =: *vent* (ou *tornade*, *courant*, etc…). Comme l'explique J. Giry-Schneider (1991), ceci semble être dû à la nature dynamique du vent.

**Cette voiture (fait + être de) 10 km/h*
Ce vent (fait + être de) 20 km/h

Certains noms comme *coût* n'entrent pas dans le schéma de phrase de base en *faire*, mais sont acceptés dans une forme réduite (*Ng* effacé).

*Cet achat (a + *fait) un coût de 30 euros*
*Cet achat (*a + fait) 30 euros*

Certains noms comme *vitesse* ne sont effaçables que si l'on rajoute le déterminant partitif *du* :

*Le bus fait (*E + du) 35 km/h*

Les deux structures (avec et sans déterminant partitif) sont acceptables avec les noms *intensité électrique* et *tension électrique* :

La ligne fait (E + du) 220V
La ligne fait (E + du) 2A

Par contre, l'ajout du déterminant partitif *du* n'est pas valable pour tous les noms :

** La corde fait du 10 m.*

Le verbe *mesurer* peut aussi être utilisé à la place du verbe support ; l'acceptabilité de la structure engendrée dépend aussi du *Ng* effacé.

La corde (fait + mesure) 10 m
*La ligne (fait + *mesure) 220 V*

Il est parfois possible d'ajouter un adjectif non prédicatif entre *Dnum* et *Unité* pour nuancer une mesure objective. Cet adjectif a un peu le même rôle sémantique qu'un prédéterminant.

La corde fait cinq (*petits mètres + *mètres qui sont petits*)

3.3.3.6 Autres

Revenons maintenant à notre phrase de base *N*0 *avoir un Ng de Dnum Unité*. Comme nous l'avons dit au début de la section, elle peut s'analyser à partir de deux phrases élémentaires : *N*0 *avoir un Ng* et *Ce Ng être de Dnum Unité*. En réduisant la première phrase au groupe nominal *le Ng de N0* (=: *la longueur de la piscine*) que nous substituons à *ce Ng* dans la deuxième phrase, nous obtenons une autre structure équivalente à notre structure de base :

(N0 avoir un Ng ; ce Ng être de Dnum Unité)
= Le Ng de N0 être de Dnum Unité

La longueur du chemin est de 100 m
= Sa longueur est de 100 m

Pour finir, si nous utilisons la notion d'opérateur à lien[42] de M. Gross (1981), nous avons une équivalence entre les phrases suivantes

La longueur du chemin être de 100 m

[42] Exemple : *La sœur de Léa est malade = Léa a sa sœur qui est malade = Léa a sa sœur malade*

? Le chemin a sa longueur qui est de 100 m

Par ailleurs, on observe :

Le chemin a pour longueur 100 m

Nous synthétisons toutes les structures équivalentes dans lesquelles peuvent rentrer les noms *Ng* dans le graphe théorique **N0AvoirUnNDeDnumUnite** ci dessous[43].

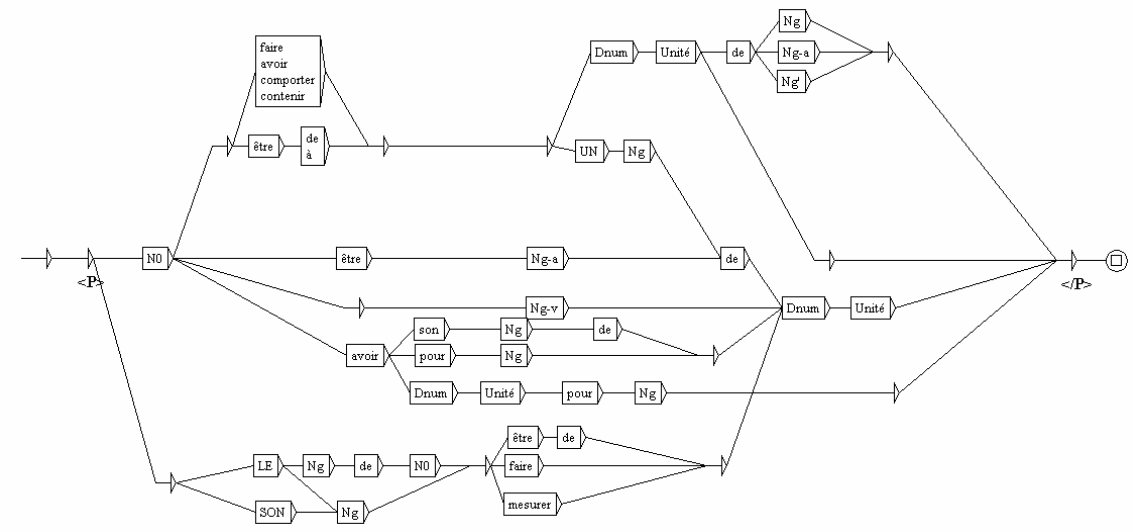

Figure 51 : N0AvoirUnNDeDnumUnite

3.3.4 Codage des propriétés

Notre analyse linguistique a montré l'existence de nombreuses contraintes dépendant des noms *Ng*. Nous souhaitons maintenant coder ces contraintes dans une table syntaxique. Nous nous inspirons de J. Giry-Schneider (1991).

Dans notre étude, les entrées lexicales correspondent aux prédicats nominaux *Ng* rentrant dans le schéma de phrase *N0 avoir un Ng de Dnum Unité*. Les propriétés que nous voulons coder sont les suivantes :

- la distribution du sujet
- la valeur lexicale des noms (*longueur, tension artérielle*)
- la contrainte entre *Ng* (ex : *largeur*) et *Unité* (ex : *mètre*)
- la variation lexicale du verbe support : *avoir, être (de + à), comporter,* etc.
- la permutation entre la séquence *Dnum Unité* et le nom *Ng*, plus les accidents morphologiques
- les transformations de nominalisation et d'adjectivation
- l'effacement du nom *N*

La table syntaxique codée est donnée plus bas et a le nom **ANMesure**. Les structures codées sont explicitement données dans l'intitulé des colonnes.

[43] Ce graphe n'est pas directement applicable à un texte (cf. section application à des textes)

Distribution du sujet

Les trois premières colonnes contiennent le codage de la distribution du sujet : *Nhum*, *Nconc* et *Npred*.

Valeur des noms

Nous indiquons la forme de base de nos prédicats nominaux *Ng*. Dans le cas des noms composés, le nom et l'adjectif apparaissent dans deux colonnes séparées.

Remarque : nous n'avons codé que partiellement le comportement de *taille de mémoire vive* pour éviter d'ajouter trop de colonnes à notre table. Nous n'avons entré que la variante réduite *mémoire vive* de type *NA*.

Contrainte Ng - Unité

Pour chaque entrée nous associons un ensemble d'unités appropriées sous la forme d'un nom de graphe (précédé de *:*). Si l'unité est vide, nous insérons le mot vide <*E*>.

Variation lexicale du verbe support

Les colonnes G, H et I correspondent respectivement aux emplois de *avoir*, *faire* et *comporter*, comme verbes supports. Nous codons aussi la propriété *il faire un Ng de Dnum Unité Loc N*0 dans la colonne J.

Remarque : nous n'avons codé qu'un certain nombre de verbes supports, seulement pour montrer que leur variation dépendait de *Ng*, comme l'avait fait J. Giry-Schneider sur un nombre de *Ng* plus réduit. Un codage complet n'est pas forcément intéressant pour l'instant car nous sommes dans un cadre assez théorique (cf. partie *réduction de la structure de base*).

La permutation de la séquence Dnum Unité et le nom Ng

L'adjectif dérivé de *Ng* est donné dans la colonne K. Le nom *Ng'* (accident morphologique du *Ng* lors de la permutation) est mis dans la colonne N. Les trois structures engendrées par les permutations sont dans les colonnes L, M et O.

Nominalisation et adjectivation

Le verbe morphologiquement et sémantiquement lié à *Ng* est donné dans la colonne Q. La possibilité d'avoir des structures à prédicat adjectival et verbal est codée dans la colonne P, R et S.

Effacement du prédicat nominal N

Nous avons codé la possibilité d'effacer *Ng* en faisant varier le verbe support et sa préposition associée : *être de* (T), *être à* (U), *faire* (V et W), *contenir* (X et Y), *compter* (Z). Pour les verbes *faire* et *contenir*, nous avons regarder la possibilité d'avoir le déterminant partitif *du* devant la séquence *Dnum Unité*.

Remarque : pour *vitesse*, comme l'effacement du Ng ne fonctionne que pour une classe sémantique de sujets extrêmement réduite (*vent, tornade*, etc.), nous mettons un signe '-' dans la case correspondante à cette propriété et à cette entrée. Nous construisons par ailleurs une grammaire locale pour cette classe à l'aide notamment du dictionnaire en-ligne des synonymes de CRISCO (http://elsap1.unicaen.fr/cherches.html). Nous reviendrons sur cette grammaire ultérieurement.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA
N0 =: Nhum	N0 =: Nconc	N0 =: Npred	Ng	Modif Appr	classe d'unités	N0 avoir un Ng de Dnum Unité	N0 faire un Ng de Dnum Unité	N0 comporter un Ng de Dnum Unité	Il faire un Ng de Dnum Unité (Loc N0+E)	Ng-a	Ng permuté	Ng-a permuté (accident)	Ng'	Ng' permuté	N0 être Ng-a de Dnum Unité	Ng-v	N0 Ng-v Dnum Unité	Dnum Unité Ng-v N0	N0 être de Dnum Unité	N0 être à Dnum Unité	N0 faire Dnum Unité	N0 faire du Dnum Unité	N0 contenir Dnum Unité	N0 contenir du Dnum Unité	N0 compter Dnum Unité	N0 avoir Dnum Unité
+	+	+	âge	<E>	:GNmesure-temps	+	-	-	-	âgé	+	-	-	-	+	-	-	-	-	-	-	-	-	-	+	+
+	+	-	aire	<E>	:GNmesure-surface	+	-	-	-	-	-	-	-	-	-	-	-	-	+	-	+	-	-	-	-	-
+	+	-	allure	<E>	:GNmesure-vitesse	+	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	+	-	-	-	-
+	+	-	capacité	<E>	:GNmesure-volume	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
+	+	+	coût	<E>	:GNmesure-monetaire	+	-	-	-	-	-	-	-	-	-	coûter	+	-	-	+	+	-	-	-	-	-
-	+	-	densité	démographique	:GNmesure-densite-pop	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	+	+
+	+	-	diamètre	<E>	:GNmesure-longueur	+	-	+	-	-	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	+	durée	<E>	:GNmesure-temps	+	-	-	-	-	-	-	-	-	-	durer	+	-	+	-	+	-	-	-	-	-
+	+	-	énergie	<E>	:GNmesure-energie	+	-	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	-	+
+	+	-	épaisseur	<E>	:GNmesure-longueur	+	+	-	-	épais	+	-	-	-	+	-	-	-	-	-	-	-	-	-	-	-
+	+	-	force	<E>	:GNmesure-force	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
+	+	-	fréquence	<E>	:GNmesure-frequence	+	+	-	-	-	-	-	-	-	-	-	-	-	+	-	+	+	-	-	-	-
+	+	-	hauteur	<E>	:GNmesure-longueur	+	+	-	-	haut	+	+	-	-	+	-	-	-	+	-	+	-	-	-	-	-
+	+	-	intensité	électrique	:GNmesure-intensite-elec	+	+	-	-	-	-	-	-	-	-	-	-	-	+	-	+	+	-	+	-	-
+	+	-	largeur	<E>	:GNmesure-longueur	+	+	-	-	large	+	+	-	-	+	-	-	-	-	-	-	-	-	-	-	-
+	+	-	longueur	<E>	:GNmesure-longueur	+	+	-	-	long	+	+	-	-	+	-	-	-	+	-	+	-	-	-	-	-
-	-	+	longueur	<E>	:GNmesure-temps	+	+	-	-	long	-	-	-	-	+	-	-	-	+	-	+	-	-	-	-	-
+	+	-	masse	<E>	:GNmesure-masse	+	+	-	-	-	-	-	-	-	-	-	-	-	+	-	+	-	-	-	-	-
-	+	-	mémoire	vive+virtuelle+cache	:GNmesure-informatique	+	-	+	-	-	+	-	-	-	-	-	-	-	+	-	+	-	+	-	-	+
+	+	-	périmètre	<E>	:GNmesure-longueur	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
+	+	-	périmètre	<E>	:GNmesure-surface	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
+	+	-	poids	<E>	:GNmesure-masse	+	+	-	-	-	-	-	-	-	-	peser	+	-	+	-	+	-	-	-	-	-
+	+	-	poids	<E>	:GNmesure-force	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-	-
+	-	-	pointure	de <pied>+de <chaussure>	<E>	+	+	-	-	-	+	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-
-	+	-	population	<E>	:GNmesure-population	+	-	+	-	-	-	-	-	-	-	peupler	-	+	-	-	-	-	+	-	+	+
-	+	-	pression	atmosphérique	:GNmesure-pression	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	+	-	profondeur	<E>	:GNmesure-longueur	+	+	-	-	profond	+	-	fond	+	+	-	-	-	-	-	+	-	-	-	-	-
+	+	-	puissance	énergétique	:GN-mesure-puissance	+	+	-	-	puissant	+	-	-	-	-	-	-	-	+	-	+	+	-	-	-	+
+	+	-	rayon	<E>	:GNmesure-longueur	+	+	+	-	-	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
+	+	-	superficie	<E>	:GNmesure-surface	+	+	+	-	-	+	-	-	-	-	-	-	-	+	-	+	-	-	-	-	-
+	+	-	surface	<E>	:GNmesure-surface	+	+	+	-	-	+	-	-	-	-	-	-	-	+	-	+	-	-	-	-	-
+	+	-	taille	<E>	:GNmesure-longueur	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-	-
+	-	-	taille	de <pied>+de <chaussure>	<E>	+	+	-	-	-	+	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-
+	+	-	température	<E>	:GNmesure-temperature	+	+	-	+	-	-	-	-	-	-	-	-	-	+	+	-	-	-	-	-	-
+	+	-	tension	électrique	:GNmesure-tension	+	+	-	-	-	-	-	-	-	-	-	-	-	+	+	+	+	-	+	-	-
+	-	-	tension	artérielle	<E>	+	-	-	-	-	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
+	+	-	vitesse	<E>	:GNmesure-vitesse	+	-	-	-	-	-	-	-	-	-	-	-	-	-	+	-	+	-	-	-	-
+	+	-	volume	<E>	:GNmesure-volume	+	+	-	-	volumineux	-	-	-	-	+	-	-	-	-	-	+	-	+	-	-	-

Table 3 : ANMesure

3.3.5 Les prédéterminants

Nous regardons maintenant le comportement de certains prédéterminants dans notre phrase de base. Nous reprenons la liste des prédéterminants fournie dans M. Gross (1977) et étudions systématiquement ceux qui peuvent apparaître dans les structures étudiées. A travers cette liste de taille modeste, nous montrons la difficulté de traiter ces composants localement. Notre liste de prédéterminants est la suivante :

> *comme, d'abord, encore, ensuite, environ, jusqu'à, même, ne…que, plutôt, presque, quelque, seul, à demi, à peine, au mieux, approximativement* et *pas tout à fait*

Nous avons travaillé sur les structures *N*0 *avoir un Ng de Dnum Unité* et *N*0 *faire Dnum Unité* et nous avons essayé d'insérer systématiquement ces prédéterminants à différents endroits. Nous avons constaté une grande variation de comportement selon les prédéterminants. Nous avons testé pour chaque prédéterminant *Préd* l'acceptabilité des sept structures suivantes :

(1) N0 avoir un Ng de Préd Dnum Unité
=: Ce bateau a une longueur d'environ 100 m

(2) N0 faire Préd Dnum Unité
=: Ses appartements font jusqu'à 50 mètres carrés

(3) N0 avoir un Ng de Dnum Unité Préd
=: Luc a un poids de 60kg à peine

(4) N0 faire Dnum Unité Préd
=: Cette ligne fait 110 V approximativement

(5) Préd N0 avoir un Ng de Dnum Unité
=: Au mieux mon moteur a une fréquence de dix tours par minute

(6) N0 avoir Préd un Ng de Dnum Unité
=: Cette ville maudite a encore une population de 100 habitants.

(7) N0 avoir un Ng Préd de Dnum Unité
=: Son spectacle n'a une durée que de dix minutes

Après examen exhaustif, nous constatons que seuls trois prédéterminants peuvent s'insérer n'importe où dans la phrase : *approximativement, au mieux* et *plutôt.* Ils jouent clairement des rôles d'adverbes. D'autres ont quelques restrictions comme les prédéterminants *à peine* ou *environ* dans :

**(A peine + Environ) Luc a une taille de 1,50 m.*
*Luc a une taille (*à peine + ?environ) de 1,50 m*

La structure non-connexe *ne … que* s'emploie sans difficulté sauf dans les cas clairs suivants :

** Que son spectacle n'a une durée de dix minutes*
** Son spectacle n'a une durée de que dix minutes*
** Son spectacle n'a une durée de dix minutes que*

D'autres ne s'emploient jamais tels que *seul* et *à demi*.

Nous construisons une table syntaxique représentant les comportements des prédéterminants. Chaque ligne correspond à un prédéterminant de notre lexique. La première colonne contient les prédéterminants. Les sept autres colonnes correspondent aux sept structures ci-dessus. Le signe '+' dans une case signifie que l'entrée correspondant à la ligne de cette case rentre dans la structure associée à sa colonne. Le signe '-' signifie le contraire.

Préd	N0 avoir un Ng de Préd Dnum Unité	N0 faire Préd Dnum Unité	N0 avoir un Ng de Dnum Unité Préd	N0 faire Dnum Unité Préd	Préd N0 avoir un Ng de Dnum Unité	N0 avoir Préd un Ng de Dnum Unité	N0 avoir un Ng Préd de Dnum Unité
comme	-	-	-	-	-	-	-
d'abord	-	-	-	-	+	+	-
encore	-	+	+	+	+	+	-
ensuite	-	-	-	-	+	+	-
environ	+	+	+	+	-	+	+
jusqu'à	-	+	-	-	-	+	-
même	-	+	-	-	-	+	-
ne...que	-	+	-	-	-	+	+
plutôt	+	+	+	+	+	+	+
presque	+	+	-	-	-	+	-
quelque	+	+	-	-	-	-	-
seul	-	-	-	-	-	-	-
à demi	-	-	-	-	-	-	-
à peine	+	+	+	+	-	+	-
au mieux	+	+	+	+	+	+	+
approximativement	+	+	+	+	+	+	+
pas tout à fait	+	+	-	-	-	+	-

Table 4 : Pred

Ainsi, pour décrire précisément les phrases de mesure en tenant compte des prédéterminants, il faut construire un graphe de prédéterminants pour chaque point potentiel d'insertion dans la structure, à l'aide de notre table syntaxique. Notre liste ne contenant qu'une petite partie des prédéterminants, notre travail est incomplet et ne permet pas de décrire précisément le comportement de tous les prédéterminants dans les phrases de mesure. Il confirme que l'on ne peut pas tenir compte des prédéterminants dans les phrases de mesure si l'on ne regarde pas la phrase complète (cf. M. Gross, 1977).

3.3.6 Réduction de la phrase élémentaire

Les phrases que nous avons étudiées dans la partie précédente sont très théoriques car elles apparaissent très peu telles quelles dans les textes. En fait, elles se retrouvent sous la forme de groupes nominaux qui sont des réductions de ces phrases. Dans cette partie, nous décrivons les processus linguistiques permettant de passer des phrases de base à ces séquences.
Nous partons de quatre schémas de phrases équivalents à *N0 avoir un Ng de Dnum Unité* :

*(a) N*0 *avoir un Ng de Dnum Unité*
*(b) N*0 *être Prep un Ng de Dnum Unité*
*(c) N*0 *être Ng-a de Dnum Unité*
*(d) Le Ng de N*0 *être de Dnum Unité*

Tout d'abord, on peut voir la structure (a) comme équivalente à *N*0 *avoir N*1 (=: *Max a un ballon*) qui se réduit en :

N1 (qu'avoir + de) N0 =: le ballon (qu'a + de) Max

Ainsi, on a :

La corde a une longueur de 10 cm
la longueur de 10 cm de la corde (réduction)

La salle a une température de 10°C
La température de 10°C de la salle (réduction)

Cette propriété a un diamètre de 14 km
Le diamètre de 14 km de la propriété (réduction)

Le spectacle a une durée de 10 min
La durée de 10 min du spectacle (réduction)

Lorsque le nom *Ng* est effacé (cf. précédemment), le déterminant est toujours *les.*

*(les + *E)10 cm de la corde*
*(les + *E) 10°C de la salle*
*(les + *E) 14 km de la propriété*
*(les + *E) 10 min du spectacle*

Pour certains *Ng*, les deux déterminants peuvent être indéfinis :

une longueur de 10 cm de (E + la) corde*
**une température de 10°C de (E + la) salle*

Le nom *Ng* peut souvent s'effacer ; dans ce cas, le premier déterminant peut être défini :

*(les + E) 10 cm de (E + *la) corde*

De plus, dans cette construction, *Det Ng de Dnum Unité de* a le statut des déterminants nominaux étudiés par P.A. Buvet (1993,1994). Le nombre de classes de noms prédicatifs *Ng* rentrant dans cette structure est limité :

- *volume, capacité* (*GNmesure-volume*)
- *longueur* (*GNmesure-longueur*)
- *durée, longueur* (*GNmesure-temps*)
- *coût, prix* (*GNmesure-monnaie*)
- *surface, superficie, aire* (*GNmesure-surface*)
- *poids, masse* (*GNmesure-masse*)

Du fait de différences de sens qui peuvent être importantes, nous représentons ces séquences dans une autre grammaire que celle décrivant les réductions nominales de nos phrases de base. Nous tenons compte uniquement des formes dans lesquelles le *Ng* est effacé du type *(E + les) 10 cm de corde*. Nous donnons ci-dessous le graphe (**DnumUniteDe**) répertoriant l'ensemble de ce type de déterminants nominaux :

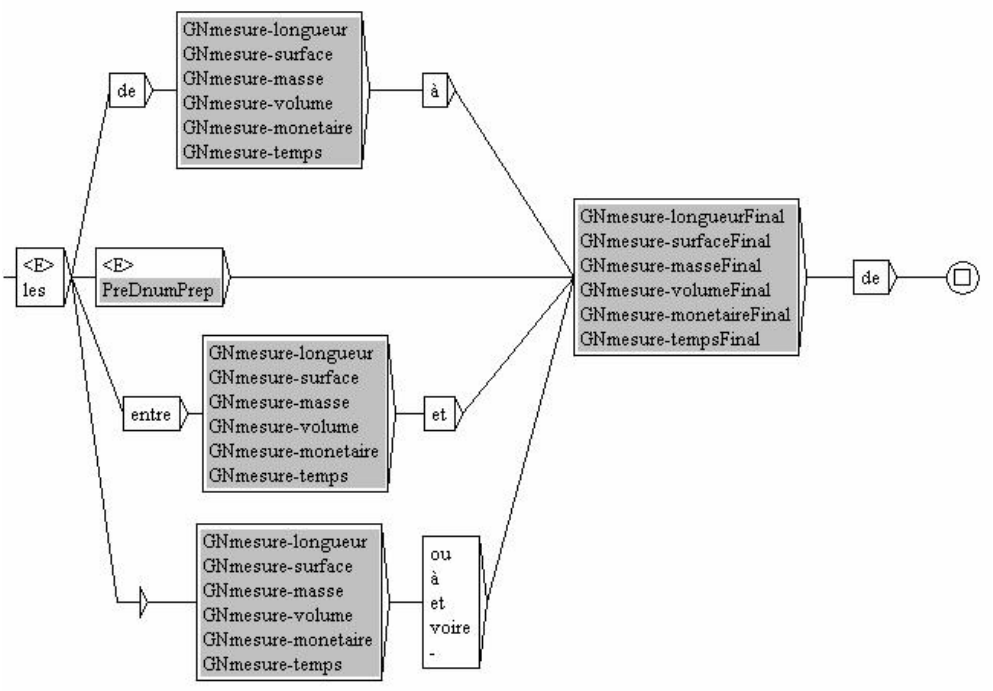

Figure 52 : DnumUniteDe

La structure (b) qui correspond au schéma de phrase *N*0 *être Prep N*1 donne lieu à une relative sujette à la réduction suivante :

*N*0 *(qui être+ E) Prep N*1
=: *Ce projet (qui est + E) de grande envergure*
=: *Cet homme (qui est + E) à la rue*

*N*0 *(qui être + E) Prep un Ng de Dnum Unité*
=: *la corde (qui est + E)d'une longueur de 10 mètres*
=: *L'eau (qui est + E) à une température de 100°C*

L'effacement du nom prédicatif *Ng* dans la forme réduite est possible lorsqu'il est possible dans la phrase de base :

Son tuyau est d'une longueur de 10 m
Son tuyau est de 10 m
Son tuyau de 10 m

L'eau est à une température de 50 degrés
L'eau est à 50 degrés
L'eau à 50 degrés

Comme on l'a déjà noté, le nom *vitesse* a un comportement particulier. Il n'accepte pas d'effacement sauf pour les *N*0 de la classe des vents :

** un bus de 40 km/h*
un (vent + courant) de 30 nœuds

Les phrases en *être à* génèrent des groupes prépositionnels adverbiaux : *à 10 km/h*. Ce verbe support a pour variantes certains verbes qui dépendent du sujet utilisé :

*(La voiture + Max + *Le vent) roule à (une vitesse de + E) 17 km/h*
*(*La voiture[44] + Max +* Le vent) court à (une vitesse de + E) 17 km/h*
*(*La voiture + ?*Max + Le vent) souffle à (une vitesse de + E) 60 noeuds*

Comme on l'indiqué précédemment, nous construisons une grammaire spécifique à la classe des vents. Nous ajoutons aussi la possibilité de reconnaître les expressions :

Un vent de force 8
Un vent de force 8 à 9

Le graphe représentant ces expressions est donné ci-dessous (**UnVentDeDnumNmesure-vitesse**). Les expressions reconnues par ce graphe sont semi-figées car la variation lexicale est faible pour *N*0 et la structure est figée.

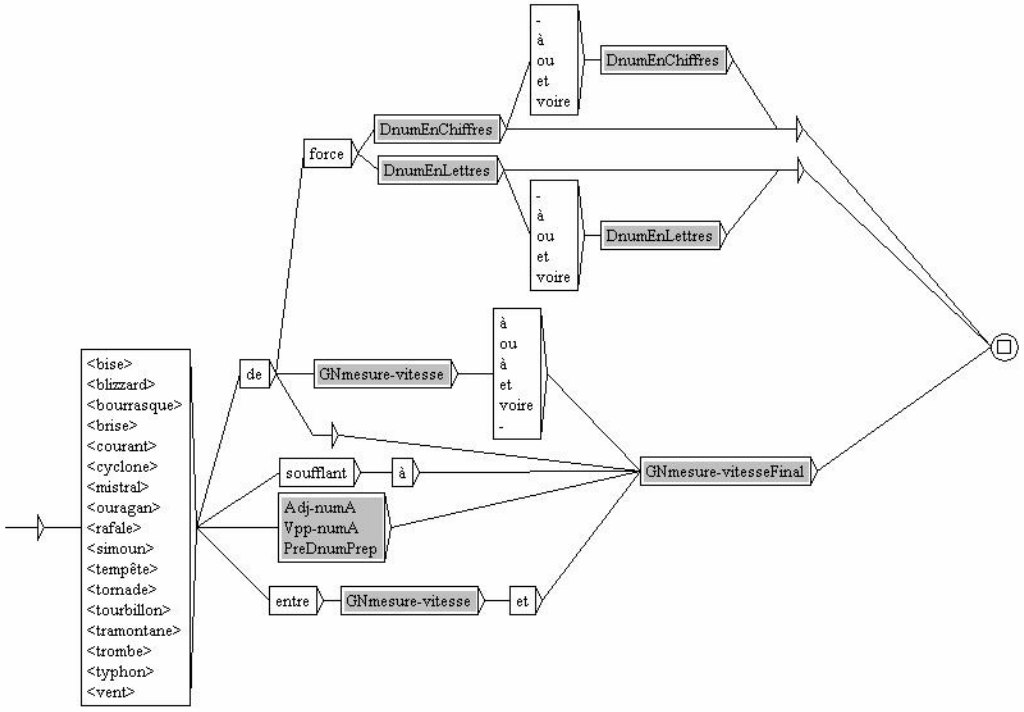

Figure 53 : VentDeDnumNmesure-vitesse

La troisième structure (c) peut également être sujette à une réduction à un groupe nominal. La partie à droite du verbe support *être* devient alors un modifieur adjectival de *N*0.

Max veut escalader une falaise (qui est + E) haute de 120 m

La dernière structure (d) peut aussi être réduite, même si cela paraît peu naturel :

La hauteur de la falaise est de 100 m
?La hauteur de la falaise de 100 m

[44] Ce nom peut être accepté si on imagine que l'on a un homme déguisé en voiture.

La pronominalisation du *N*0 la rend plus naturelle :

Sa hauteur de 100 m m'effraie

Nous proposons ci-dessous le graphe paramétré décrivant l'ensemble des formes réduites de notre phrase de base. Ces formes réduites sont toutes des groupes nominaux. Pour l'entrée *largeur*, nous générons le graphe associé. La variable *@D* est remplacée par l'entrée lexicale *largeur*. La variable *@L* correspondant à la propriété de permutation entre *Dnum Unité* et *Ng* est remplacée par le mot vide (<*E*>) car cette propriété est autorisée pour cette entrée (+). Par contre, la boîte contenant *@U* est supprimée car la structure correspondante (*N*0 *être à Dnum Unité*) est interdite (symbole -), etc.

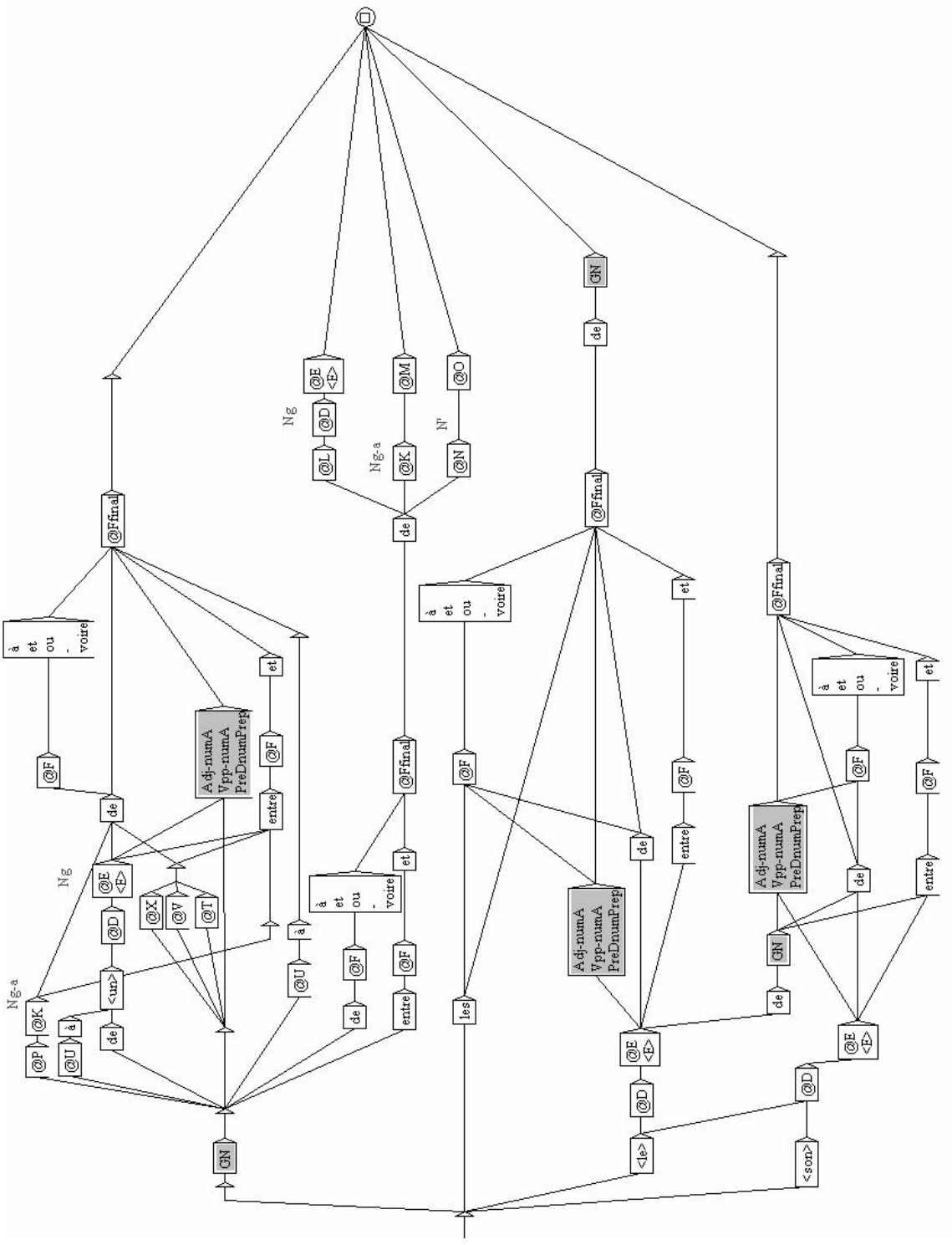

Figure 54 : graphe patron décrivant les réductions de N0 avoir un Ng de Dnum Unite

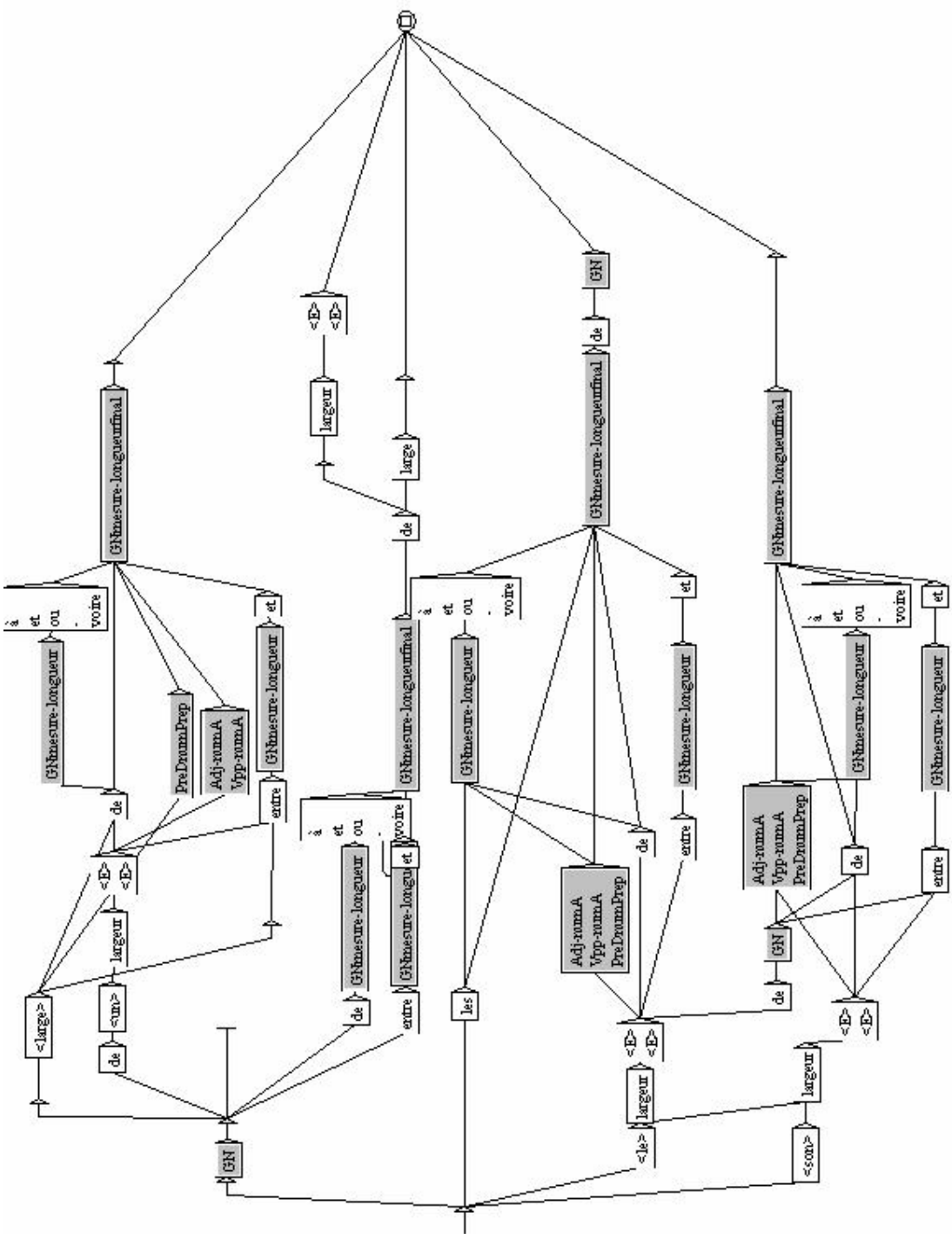

Figure 55 : graphe généré pour l'entrée *largeur*

3.4 Représentation des mesures relatives

3.4.1 Généralités

Dans cette section, nous regardons des structures mesurant une caractéristique ou propriété (*Ng*) d'un élément *N*0 par rapport à un autre (*N*1). Les schémas de phrase étudiés sont de la forme :

*N*0 *Vsup Prép un Ng de Dnum Unité Prép N*1
=: *Paris est à une distance de quelques milliers de kilomètres de New York*
=: Max est dans un rayon de 50 km autour de Reims
=: *Le stylo forme un angle de 90° avec la règle*

Nous regardons d'abord le comportement syntaxique des prédicats nominaux *Ng* rentrant dans cette structure. Puis, nous montrons que les expressions de pourcentage peuvent aussi être classées dans cette catégorie et nous examinons leurs propriétés distributionnelles. Enfin, nous étudions certaines expressions exprimant une comparaison relative de mesure.

3.4.2 Etude de la structure N0 Vsup Prép un Ng de Dnum Unité Prép N1

L'ensemble des prédicats *Ng* rentrant dans cette structure est restreint :

altitude, angle, distance, hauteur, périmètre, profondeur, rayon

Nous constatons qu'ils désignent tous des caractéristiques « géométriques ». Le nom *altitude* semble particulier car la phrase ci-dessous est un peu bancale :

?L'avion est à une altitude de 10 000 m au dessus du niveau de la mer

En effet, on ressent une certaine redondance avec l'utilisation de *au dessus de la mer* car cette séquence se trouve implicitement dans le nom *altitude* qui est la distance verticale entre le niveau de la mer et l'élément dont on veut mesurer l'altitude. Cependant, l'effacement de *altitude* force la présence de cette séquence.

*L'avion est à 10 000 m (?*E + au dessus du niveau de la mer)*

Comme dans la structure précédente exprimant une mesure « absolue », chaque *Ng* sélectionne un ensemble d'unités comme montré dans le tableau ci dessous.

altitude	GNmesure-longueur	*5 mètres*
angle	GNmesure-angle	*360 °*
distance	GNmesure-longueur	*trois kilomètres*
	GNmesure-temps	*douze secondes*
hauteur	GNmesure-longueur	*13 cm*
périmètre	GNmesure-longueur	*1 centimètre*
	GNmesure-surface	*78 m2*
profondeur	GNmesure-longueur	*dix mètres*
rayon	GNmesure-longueur	*5 km*

Table 5 : contrainte entre *Ng* et *Unité*

On constate l'apparition d'une nouvelle classe d'unités *GNmesure-angle* qui comporte les unités mesurant un angle tels que *radian, degré* (*rad*, °). Notons le cas particulier de *distance* qui sélectionne à la fois des unités de mesure de longueur (*Nmesure-longueur*) et des unités de mesure de temps (*Nmesure-temps*) :

Paul est à une distance de (?10 min + 10 km) de la maison

Cette forme est peu naturelle avec les unités de temps, mais l'emploi de ces unités le devient tout à fait lorsque l'on efface le prédicat *distance* :

Paul est à (10 min + 10 km) de la maison

Les noms *hauteur* et *profondeur* entrent aussi dans le schéma de phrase exprimant une mesure « absolue ». Cependant, l'emploi absolu et l'emploi relatif sont bien distincts :

Paul est à une hauteur de 10 m (E + au dessus du sol)
** Paul a une hauteur de 10 m*

*L'immeuble a une hauteur de 100 m (E + *au dessus du sol)*
**L'immeuble est à une hauteur de 100 m*

Dans le premier ensemble de phrases, le nom *hauteur* désigne la distance verticale entre Paul et le sol. Dans le deuxième ensemble de phrases, il désigne une caractéristique intrinsèque de l'immeuble du même type que *largeur* ou *longueur*.

Nous pouvons diviser cet ensemble de noms en trois. En effet, ils entrent dans trois structures bien distinctes :

(a) *N0 avoir un Ng de Dnum Unité avec N1*
=: *le crayon a un angle de 45° avec le livre*

(b) *N0 être à un Ng de Dnum Unité (de + Loc) N1*
=: *Le plongeur est à une profondeur de 10 m sous l'eau*
=: *Marie est à une distance de trois kilomètres de Paris*

(c) *N0 être dans un Ng de Dnum Unité autour de N1*
=: Les soldats sont dans un rayon de 100 km autour de la ville

Le seul nom prédicatif *Ng* rentrant dans le schéma de phrase (a) est *angle*. Les phrases en *être à* sont interdites et les verbes supports les plus naturels sont *faire* et *former*.

*Le livre (a + fait + forme + *est à) un angle de 45° avec le crayon*

Dans le cas général, la structure *N*0 *avoir un Ng avec N*1 est symétrique et se réduit à la forme nominale en *entre …et …*, comme dans l'exemple ci-dessous :

La France a une frontière avec la Belgique
La Belgique a une frontière avec la France
La France et la Belgique ont une frontière (?E + commune)
=> la frontière de la France avec la Belgique (réduction)

la frontière entre la France et la Belgique (réduction)

Dans notre cas, *angle* se comporte de la même manière, même si mathématiquement la symétrie n'est pas vérifiée car un angle est signé. En effet, on peut analyser (a) comme deux phrases :

Le livre a un angle de 45° avec le crayon
= Le livre a un certain angle avec le crayon ; cet angle est de 45°

Ainsi, on retrouve le cas général dans la première phrase que l'on peut réduire à *l'angle entre le livre et le crayon.* On obtient les phrases équivalentes suivantes :

Le livre a un angle de 45° avec le crayon
= L'angle du crayon avec le livre est de 45°
= L'angle entre le crayon et le livre est de 45°

L'emploi du verbe support *avoir* est également possible avec le nom *altitude* :

?L'avion a une altitude (E + de croisière) de 10 000 m (E + au dessus de la mer)

Cependant, *altitude* ne rentre pas dans la même structure de base que *angle* :

**Ce pic a une altitude de 3 290 m avec le niveau de la mer*

Nous examinons maintenant la structure (b) en *être à*. Les noms entrant dans ce schéma de phrase sont : *altitude*, *distance*, *hauteur*, *profondeur*. Ces quatre entrées ont toutes un comportement propre. Tout d'abord, seuls les noms *distance* et *hauteur* ont un comportement symétrique, bien qu'ils n'entrent pas dans le schéma de phrase (a). Cela n'est pas étonnant pour *distance* du fait de sa définition mathématique. Pour le nom *hauteur*, c'est moins net comme le montrent les phrases un peu bancales ci-dessous.

*Max (*a + est à) une distance de 10 m (de + *avec) Marie*
Max et Marie sont à une distance de 10 m (l'un de l'autre + E)
La distance entre Max et Marie est de 10 m

La hauteur entre l'avion et le toit de la maison est de 15 m
? l'avion et le toit de la maison sont à une hauteur de 15 m (l'un de l'autre + E)

Seul le nom *distance* peut être sujet à une transformation d'adjectivation. L'adjectif morphologiquement lié (*Ng- a*) à *distance* est *distant* :

Paul est distant de 15 m de Max
Paul et Max sont distants de 15m (E + l'un de l'autre)

Même des noms comme *profondeur* et *hauteur* possédant un *Ng- a* ne sont pas sujets à cette transformation :

** Paul est haut de 15 m du sol*
** Le plongeur est profond de 50 m sous l'eau*

Pour tous les noms rentrant dans (b), la séquence *Dnum Unité* peut être permutée avec *Ng* ; de même, la séquence *de N*1 est effaçable selon le contexte :

Max est à deux mètres de distance (E + du mur)
Marie est à 2 m de hauteur (E + du sol)
L'explorateur est à 200 mètres de profondeur (E + du niveau du sol)
L'avion est à 10 000 m d'altitude (E + ? au-dessus du niveau de la mer)

On observe couramment dans des textes la présence de formes réduites de la structure très théorique *N0 être à un Ng de Dnum Metre Loc N1* :

Marie est à 20 m sous l'eau
= Marie est à une profondeur de 20 m sous l'eau

Luc est (suspendu) à 2 m au-dessus du sol
= Luc est (suspendu) à une hauteur de 2 m au-dessus du sol

L'avion est à 10 000 m au-dessus du niveau de la mer
= ? L'avion est à une altitude de 10 000 m au-dessus du niveau de la mer

On peut retrouver le *Ng* effacé à partir de la préposition locative qui indique une direction. Par exemple, *sous* indique une direction verticale vers le bas et ainsi on déduit que l'on a une *profondeur*. Le *N*1 revêt une importance certaine. En effet, la préposition locative *au-dessus de* indique une direction verticale vers le haut, ce qui peut correspondre soit à une *hauteur*, soit à une *altitude*. Le nom *altitude* sélectionnant clairement un ensemble restreint d'expressions quasi-figées de la forme *Loc N1* comme *au-dessus du niveau de la mer*, il est facile de choisir entre les deux possibilités.
Cette analyse est faisable mais ne nous paraît pas très convaincante car elle est restreinte à un petit ensemble de prépositions. En effet, comment analyser la phrase suivante ?

La piste est à 10 km en aval de Val d'Isère

Nous décidons d'analyser la structure *N*0 *être à Dnum Metre Loc N*1 à l'aide des deux schémas de phrase suivants :

*N*0 *être à une distance de Dnum Metre de N*1
*N*0 *être Loc N*1 (Si *Loc* ≠ *de*)

Par ce moyen, on distingue clairement distance et direction : la première phrase indique la distance entre *N*0 et *N*1. La deuxième phrase donne la direction (optionnellement, le sens) ou une autre information géométrique pour retrouver (mathématiquement) la position de *N*0 par rapport à *N*1. Soit la phrase :

Bordeaux est à 550 km au sud-ouest de Paris

Elle s'analyse par :

Bordeaux est à une distance de 550 km de Paris ⇔ d(Bordeaux, Paris) = 550 km[45]

[45] *d*(*x*,*y*) désigne la distance entre le point *x* et le point *y*.

Bordeaux est au sud-ouest de Paris ⇔ direction = *sud-ouest*

Dans les phrases en *être à*, l'effacement de la préposition *à* est possible lorsque le *Ng* a déjà été effacé et que la préposition est autre que *de* :

Max est (E + à) 100 m sous terre
L'avion est (E + à) 10 000 m au dessus du niveau de la mer
Marie est (E + à) 100 km au nord de Marseille

*Marie est (*E + à) 100 km de Marseille*
*Max est (*E + à) 100 m du sol*
*Max est (*E + à) une distance de 200 m de la maison*

En anglais, on observe une structure de phrase équivalente à la différence près que l'on n'a pas de préposition après le verbe support :

N0 be Dnum Unit Loc N1
=: John is 30 miles (in the north of + from) London

On constate que la préposition locative *in the north of* peut être réduite à *north of* dans la phrase, ce qui rend l'expression plus compacte qu'en français :

John is 30 miles north of London
Jean est 50 km nord de Paris

Le nom *distance* accepte des modifieurs adverbiaux dans sa structure en *être à* tels que *à vol d'oiseau, à la ronde*. Son insertion dans la phrase conserve la symétrie, ce qui n'est pas le cas lorsque l'on a une préposition locative indiquant un sens et/ou une direction :

Paris est à 220 km à vol d'oiseau de Lille
Lille est à 220 km à vol d'oiseau de Paris

Lille est à 220 km au nord de Paris
** Paris est à 220 km au nord de Paris*
Paris est à 220 km au sud de Lille (équivalence sémantique et non syntaxique)

Nous constatons également une autre différence de comportement entre ces deux types de séquences si l'on analyse la phrase de base comme deux phrases élémentaires :

Paris est à une distance (de 200 km à vol d'oiseau + à vol d'oiseau de 10 km) de Lille
*Paris est à une distance de 200 km de Lille ; Cette distance est à vol d'oiseau (E + *de Lille)*

Paris est à une distance de 220 km au sud de Lille
Paris est à une distance de 220 km de Lille ; Paris est au sud de Lille

Le nom *distance*, lorsqu'il sélectionne des unités de temps, autorise l'insertion de modifieurs adverbiaux appropriés comme dans :

Le centre-ville est à dix minutes (en voiture + à pied + de marche) de Paris

Ils s'analysent à peu près de la même manière que *à vol d'oiseau*, même si l'on constate que la phrase en *être* est difficile. Pour rendre cette dernière phrase plus naturelle, il faut ajouter la forme passive du verbe *parcourir*.

Le centre-ville est à une distance de 10 minutes de Paris
Cette distance est (?E + parcourue) (en voiture+à pied)

Par ailleurs, l'analyse de *de marche* est impossible par ce moyen, ce qui n'est pas étonnant du fait que *10 minutes de marche* est équivalent à *une marche de 10 minutes*. \

**Cette distance est de marche*
Cette distance correspond à une marche d'une durée de 10 minutes

Les graphes **N0EtreADnumMetreLocN1** et **N0EtreADnumNtempsLocN1** décrivent ces structures avec le *Ng* effacé. Dans le premier, les unités sélectionnées appartiennent à la classe *GNmesure-longueur* ; dans le deuxième, elles appartiennent à *GNmesure-temps*. Les adverbes appropriés au nom *distance* sont représentés dans les graphes **Adv-app-distance1** et **Adv-app-distance2** : le premier concerne les unités de mesure de longueur et le deuxième concerne les unités de mesure de temps.

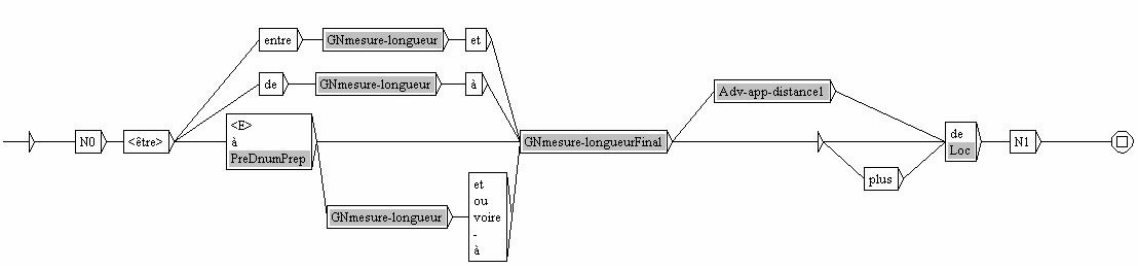

Figure 56 : N0EtreADnumMetreLocN1

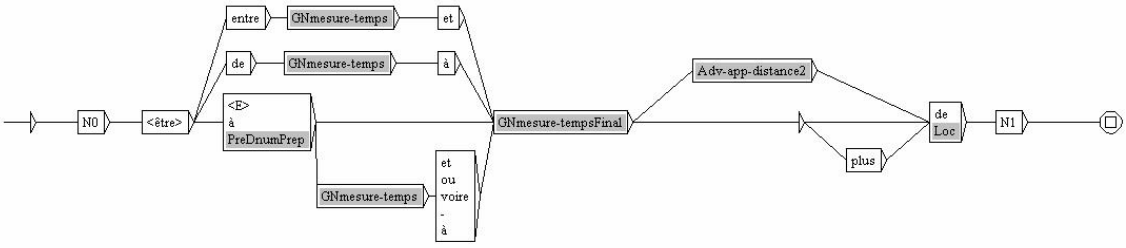

Figure 57 : N0EtreADnumNtempsLocN1

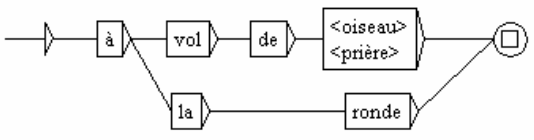

Figure 58 : Adv-app-distance1

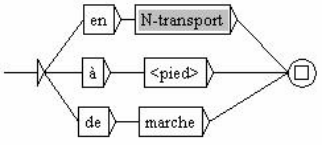

Figure 59 : Adv-app-distance2

La structure (c) diffère des deux autres par les prépositions (*Prep* =: *dans* et *prep1* =: *autour de*). Il existe deux noms rentrant dans cette structure : *périmètre* et *rayon*. Ils ont deux

comportements différents. Tout d'abord, *périmètre* sélectionne deux types d'unités : les unités de mesure de longueur (conformément à l'emploi mathématique) et les unités de mesure de surface (emploi courant). D'autre part, la phrase de base pour *rayon* est la réduction d'une phrase plus longue, ce qui n'est pas le cas pour *périmètre* :

Max est dans un rayon de 10 km autour de Lille
= Max est dans un cercle d'un rayon de 10 km autour de Lille

3.4.3 Codage des propriétés dans une table syntaxique

Bien que le nombre d'entrées lexicales soit peu élevé, le nombre de phrases à représenter dans les graphes est très important. Ainsi, nous décidons de coder les contraintes décrites précédemment dans une table syntaxique. Chaque ligne correspond à une entrée lexicale. La première colonne contient l'information indiquant si le sujet peut être un élément prédicatif. La deuxième colonne indique le verbe support employé alors que la troisième colonne donne la préposition suivant *Vsup*. La quatrième colonne comporte les entrées lexicales. La colonne 5 indique les classes d'unités sélectionnées par les entrées. Les colonnes 6 à 10 concernent la variation lexicale des prépositions *Prep1* : soit *avec*, soit *de*, soit *au-dessus de*, soit *autour de*, soit *Loc* qui désigne n'importe quelle préposition locative (simple ou composé, cf. Chapitre suivant). La onzième colonne concerne l'effacement de *Prep1 N1* dans la phrase de base. Le figement de la séquence *Prep1 N1* est codé dans la colonne 12. Le graphe **Prep1N1-altitude** représente l'ensemble des séquences figées *Prep1 N1* du nom *altitude*. Les propriétés de symétrie et de permutation sont respectivement données dans les colonnes 13 et 14. Les adverbes appropriés sont indiqués dans la colonne 15. Les colonnes 16 et 17 concernent les transformations d'adjectivation.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
N0Pred	Vsup	Prep	Ng	GNmesure	Prep1=: avec	Prep1=:de	Prep1=:à le-dessus de	Prep1=:autour de	Prep1=:Loc	N0 Vsup Prep un Ng de Dnum Unite	PrepN1-figé	permutation	symétrie	Adv-app	Ng-a	N0 être Ng-a de Dnum Unite Prep N1	N0 être dans un cercle de un Ng de Dnum Unité
-	<faire>	<E>	angle	:GNmesure-angle	+	-	-	-	-	-	-	-	+	-	-	-	-
+	<être>	à	distance	:GNmesure-longueur	-	+	-	-	-	+	-	+	+	:Adv-app-distance1	<distant>	+	-
+	<être>	à	distance	:GNmesure-temps	-	+	-	-	-	+	-	+	+	:Adv-app-distance2	<distant>	+	-
+	<être>	à	hauteur	:GNmesure-longueur	-	+	+	-	-	+	-	+	+	-	-	-	-
+	<être>	à	profondeur	:GNmesure-longueur	-	+	-	-	+	+	-	+	-	-	-	-	-
+	<être>	à	altitude	:GNmesure-longueur	-	-	-	-	-	+	:Prep1N1-altitude	+	-	-	-	-	-
+	<être>	dans	périmètre	:GNmesure-longueur	-	-	-	+	+	+	-	-	-	-	-	-	-
+	<être>	dans	périmètre	:GNmesure-surface	-	-	-	+	+	+	-	-	-	-	-	-	-
+	<être>	dans	rayon	:GNmesure-longueur	-	-	-	+	-	+	-	-	-	-	-	-	+

Table 6 : N0 Vsup Prep un Ng de Dnum Unité Prep1 N1

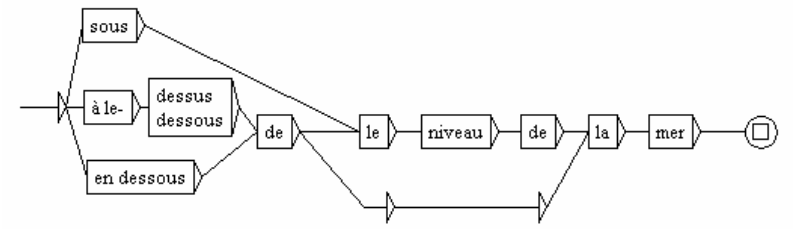

Figure 60 : Prep1 N1 figé pour le nom *altitude* (Prep1N1-altitude)

Nous présentons ci-dessous le graphe paramétré associé à la table ci-dessus, représentant l'ensemble des structures adverbiales dérivées de notre structure (ex : *à 10 m de profondeur*, *dans un rayon de trente kilomètres autour de Paris* ou *à une distance inférieure à 2m de la maison*). Juste en dessous nous donnons le graphe généré pour l'entrée *distance*.[46]

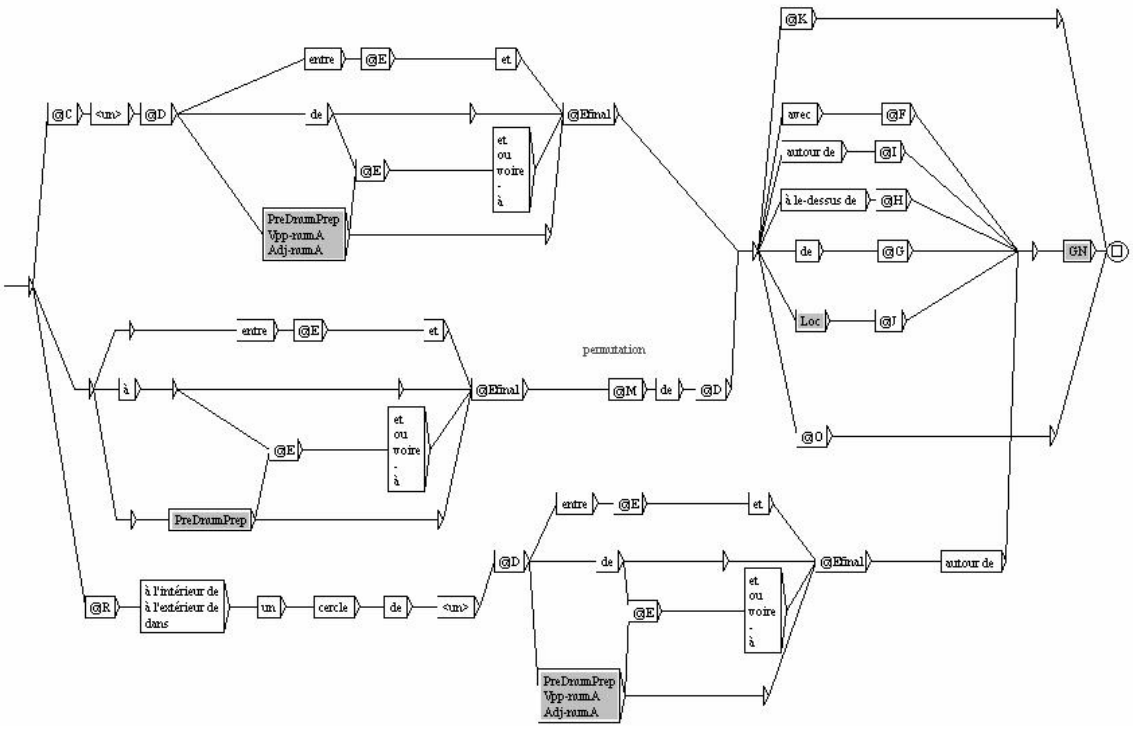

Figure 61 : graphe patron des structures adverbiales dérivées de *N0 Vsup Prep un Ng de Dnum Unite Prep1 N1*

[46] Dans le cas de l'adverbe, le graphe généré pour *angle* n'a pas lieu d'être. Nous le supprimons simplement à la main.

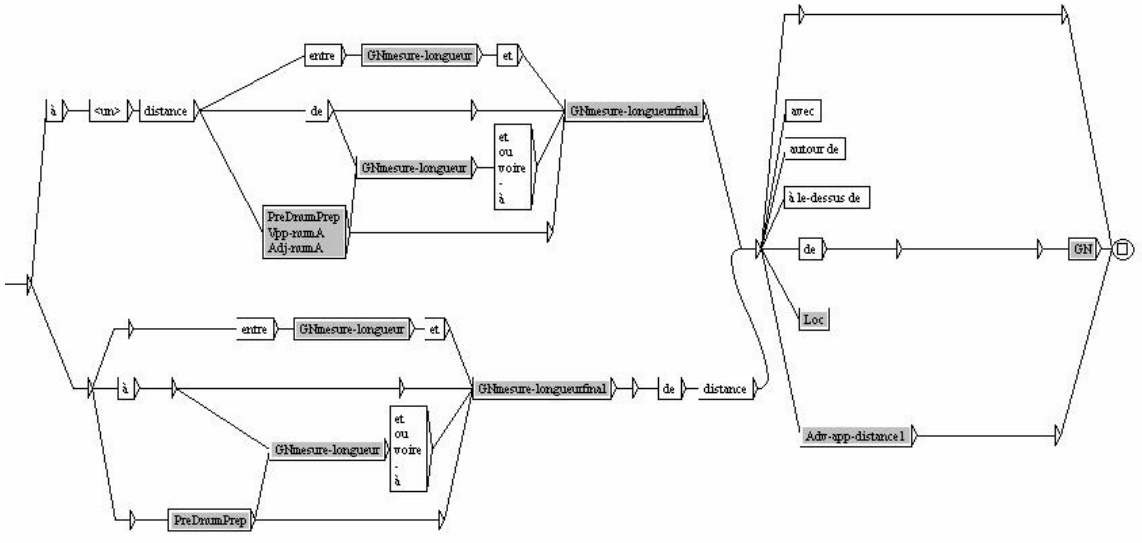

Figure 62 : graphe généré pour *distance*

3.4.4 Les expressions de pourcentage

Dans cette partie, nous nous concentrons sur les pourcentages. Nous montrons qu'ils entrent eux aussi dans le schéma de phrase *N0 Vsup Prép un Ng de Dnum Unité de N1* exprimant une mesure relative et étudions leur comportement syntaxique. Nous faisons une brève synthèse de notre étude réalisée en collaboration avec T. Nakamura (T. Nakamura et M. Constant, 2001). Cette étude montre que les expressions de pourcentages rentrent dans les structures suivantes :

N0 représenter Dnum % de N1
=: Les étudiants de Jussieu représentent 19% des étudiants parisiens

N0 comporter Dnum % de N1
=: Les étudiants parisiens comportent 19% d'étudiants de Jussieu

Ces structures sont en quelque sorte une forme réduite des structures théoriques suivantes contenant le prédicat *pourcentage*[47] :

N0 représenter un pourcentage de Dnum % de N1
=: Les étudiants de Jussieu représentent un pourcentage de 19% des étudiants parisiens

N0 comporter un pourcentage de Dnum % de N1
=: Les étudiants parisiens comportent un pourcentage de 19% d'étudiants de Jussieu

Ainsi, nous retombons bien sur notre structure de base représentant une mesure relative. Le prédicat *pourcentage* admet deux arguments (*N*0 et *N*1) et utilise les verbes supports (de pourcentage) *représenter* et *comporter*. L'unité sélectionnée est *%* (ou *pour cent* en toutes

[47] Le nom *proportion* semble également bien marcher :

Les étudiants de Jussieu représentent une proportion de 19% des étudiants parisiens

lettres). Etant donné que ces phrases sont théoriques, nous travaillons dorénavant sur les structures réduites. Ces phrases apparaissent comme les phrases élémentaires permettant d'analyser un pourcentage. Soit la phrase :

40% de (E + les) Français regardent la télé chaque soir

Cette phrase s'analyse comme suit à l'aide de deux phrases :

Des Français regardent la télévision chaque soir
(Les Français qui regardent la télévision chaque soir + Ces Français) représentent 40% des Français

Une analyse à l'aide du verbe *comporter* est également possible. La deuxième phrase deviendrait alors :

Les français comportent 40 % de personnes qui regardent la télévision chaque soir

Mais cette analyse n'est pas toujours valable. Cela dépend essentiellement de la nature sémantique (notamment le trait humain collectif) du nom tête du groupe nominal suivant la séquence *Dnum % de* :

40 % de la population regarde la télé chaque soir
**De la population regarde la télévision chaque soir*
**Cette population représente 40% de la population*

Dans ces cas-là, l'utilisation du déterminant nominal *une partie de* est préférable :

Une partie de la population regarde la télévision chaque soir
Cette partie de la population représente 40% de la population

Mais nous n'entrons pas dans la discussion. Nous souhaitons maintenant comparer les deux phrases de base des expressions de pourcentage :

Les produits laitiers représentent 80 % de notre production
Notre production comporte 80 % de produits laitiers

Ces deux phrases constituent une classe d'équivalence sémantique. En effet, ces deux phrases, qui ont exactement le même sens, ne diffèrent que par le verbe et les positions des arguments des verbes. Nous avons les équivalences suivantes :

*N*0*(représenter) = N*1*(comporter)*
*N*1*(représenter) = N*0*(comporter)*

Notons que ce schéma peut être étendu à d'autres phrases comme la phrase en *il y a* :

Il y a 80 % de produits laitiers (dans + parmi) notre production

Nous appelons complément d'inclusion d'une phrase de pourcentage l'argument *N*1 situé juste après la séquence *Dnum % de*. Le complément d'inclusion de la phrase avec *représenter* comprend obligatoirement un article défini, alors que celui du verbe *comporter* ne doit avoir

aucun article, ce qui peut correspondre à l'article indéfini par la règle de cacophonie. Ainsi, on a les quatre phrases suivantes (les deux acceptables sont équivalentes) :

Les étudiants de Jussieu représentent 19 % des étudiants parisiens
**Les étudiants de Jussieu représentent 19 % d'étudiants parisiens*
Les étudiants parisiens comportent 19% d'étudiants de Jussieu
**Les étudiants parisiens comportent 19% des étudiants de Jussieu*

Pour résumer, nous avons le schéma d'équivalence suivant :

N0 représenter Dnum % de LE N1
= N1 comporter Dnum % de ∅ N0

Notons que chacun des deux verbes est le verbe représentatif d'un ensemble de verbes ayant le même comportement syntaxique dans notre schéma de phrase : *représenter* pour l'ensemble {*représenter*, *constituer*, etc.} et *comporter* pour {*comporter*, *contenir*, *avoir*, etc.}. Pour plus de détails, se référer à T. Nakamura et M. Constant (2001).

Nous donnons ci-dessous les graphes représentant ces phrases. Les graphes **N0def, N1def** et **N0DetZ** représentent des groupes nominaux : les deux premiers comportent un déterminant défini et le dernier a un déterminant vide. **GNmesure-pourcentage** contient des unités de pourcentage (*pour cent* en toutes lettres et le symbole *%*).

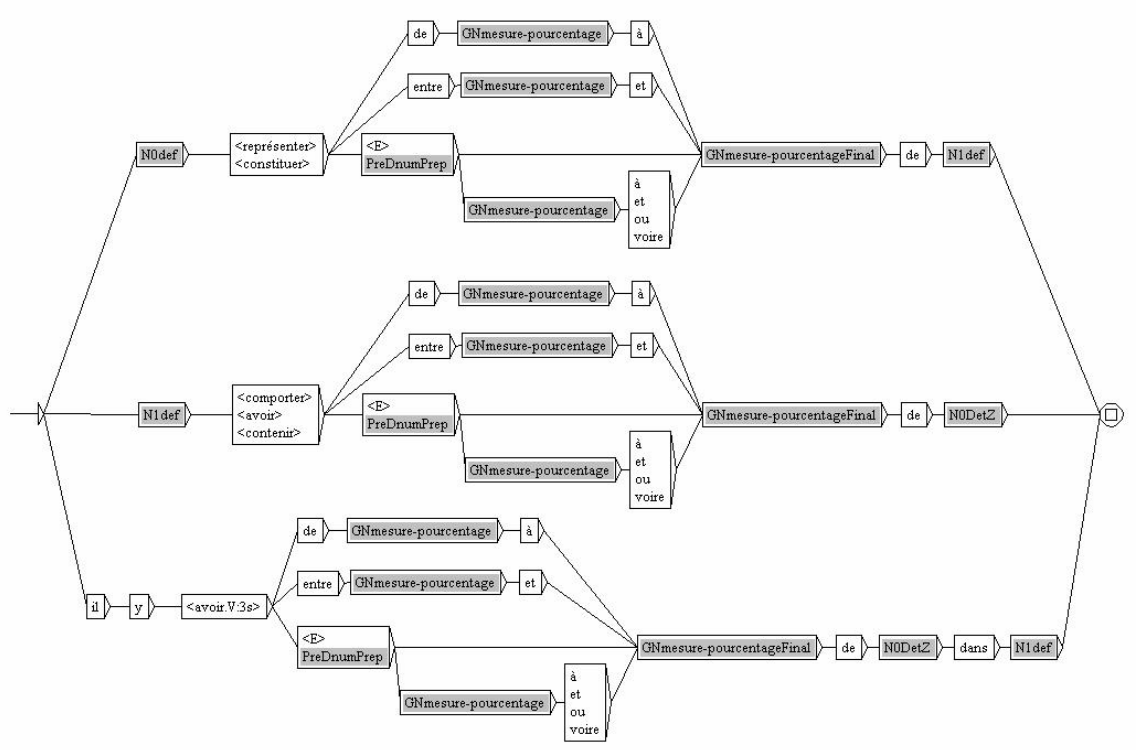

Figure 63 : N0 représenter Dnum% de N1

3.4.5 Comparaisons : quelques remarques sur les variations lexicales

Nous faisons maintenant quelques remarques sur la variation lexicale d'un nouveau type de phrases qui peuvent être considérées comme des mesures relatives. Prenons les deux phrases suivante :

Max a une taille de 178 cm
Luc a une taille de 174 cm

Il est possible d'interpréter ces phrases à l'aide d'une phrase comparative :

La taille de Max est 4 cm plus (élevée + grande) que celle de Luc
= La taille de Max dépasse de 4 cm celle de Luc

Nous nous intéressons à un autre type de structure mais qui est sémantiquement équivalente à celle-ci. La propriété mesurée n'y est plus explicite sous la forme d'un nom *Ng* (ex : *taille*) mais sous la forme implicite d'un adjectif comme dans la phrase ci-dessous équivalente à la phrase précédente :

Max est 4 cm plus grand que Luc

Ainsi, nous étudions la structure *N0 être Dnum Unité (plus+moins) Adj que N1*. Elle apparaît bien comme une mesure relative car elle met en jeu une mesure (*Dnum Unité*) et deux arguments (*N*0 et *N*1). Nous nous attachons surtout à montrer les contraintes lexicales. Nous partons des listes de noms prédicatifs *Ng* utilisées précédemment et nous regardons l'ensemble des adjectifs pouvant être utilisés dans chacun des cas. Chaque nom *Ng* sélectionne un ensemble d'adjectifs appropriés très restreint que nous pouvons facilement répertorier. Par exemple, pour le nom *poids* sélectionnant les unités de mesure de masse (*Nmesure-masse*) et pour le nom *longueur* sélectionnant les unités *Nmesure-longueur*, on a :

*Paul est 10 kg plus (léger + lourd + *grand + *chaud) que Luc*
*Ta barque est 10 m plus (petite + grande + longue + courte + *lourde + *chaude) que mon voilier*

Les noms morphologiquement dérivés de ces adjectifs (quand ils existent) ne rentrent pas, dans la plupart des cas, dans une phrase de la forme *N0 avoir un Ng de Dnum Unité* :

** Paul a une légèreté de 75 kg*
** Ta barque a une grandeur de 3 m*

Un même adjectif peut être sélectionné par plusieurs *Ng* comme *grand* sélectionné par *aire*, *hauteur*, *taille*, etc… Dans ces cas, c'est la nature sémantique des arguments *N*0 et *N*1 qui permet de lever l'ambiguïté. Nous n'entrons pas dans la discussion.

Si l'on regarde les *N*g du type *distance*, on constate le même phénomène de sélection d'adjectifs pour deux d'entre eux : *distance* et *hauteur*.

- *Ng =*: *distance*

*Paris est 300 km plus (loin + proche + *distant) de Bordeaux que de Tours*

- *Ng =*: *hauteur*

  Dans l'ascension du pic du Midi, Paul est 30 m plus (haut + bas) que Max

Dans la plupart des textes, on retrouve ces expressions sous la forme réduite d'un adverbe :

100 km plus (loin + haut), des soldats ont tiré sur des manifestants

Nous donnons dans les deux tableaux ci-dessous, les adjectifs *Adj* (lorsqu'ils existent) associés à chaque *Ng* entrant respectivement dans la structure *N0 avoir un Ng de Dnum Unité* et dans la structure *N0 Vsup Prep un Ng de Dnum Unité Prep N1* :

A	B	C
Ng	GNmesure	Adj
longueur	:GNmesure-longueur	<long>+<grand>+<petit>+<court>
profondeur	:GNmesure-longueur	<profond>
hauteur	:GNmesure-longueur	<haut>+<petit>+<bas>
largeur	:GNmesure-longueur	<large>
taille	:GNmesure-longueur	<grand>+<petit>
surface	:GNmesure-surface	<grand>+<petit>+<vaste>+<étendu>
vitesse	:GNmesure-vitesse	<vite>+<rapide>+<lent>
poids	:GNmesure-masse	<gros>+<maigre>+<lourd>+<léger>
épaisseur	:GNmesure-longueur	<épais>
température	:GNmesure-temperature	<chaud>+<froid>+<tiède>
énergie	:GNmesure-energie	<calorifique>+<énergétique>
coût	:GNmesure-monetaire	<coûteux>+<cher>+<économique>
durée	:GNmesure-temps	<rapide>+<lent>+<long>+<court>
volume	:GNmesure-volume	<volumineux>+<grand>+<petit>
puissance	:GNmesure-puissance	<puissant>+<faible>

Table 7 : entre *Ng*, *GNmesure* et *Adj* (absolu)

Ng	GNmesure	Adj
distance	:GNmesure-longueur	<loin>+<proche>+<près>
hauteur	:GNmesure-longueur	<haut>+<bas>

Table 8 : entre *Ng*, *GNmesure* et *Adj* (relatif)

Nous présentons ci-dessous le graphe patron représentant l'ensemble des expressions réduites dérivées de la structure comparative et le graphe généré pour *surface*.

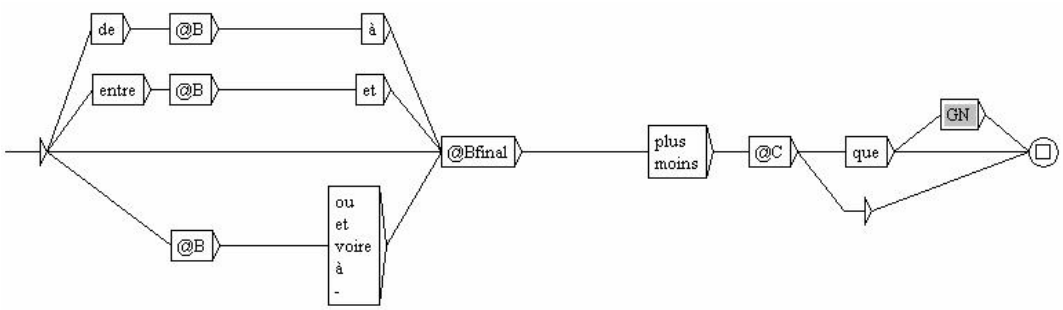

Figure 64 : graphe patron des mesures comparatives dérivées de *N0 Etre Dnum Unite plus Adj que N1*

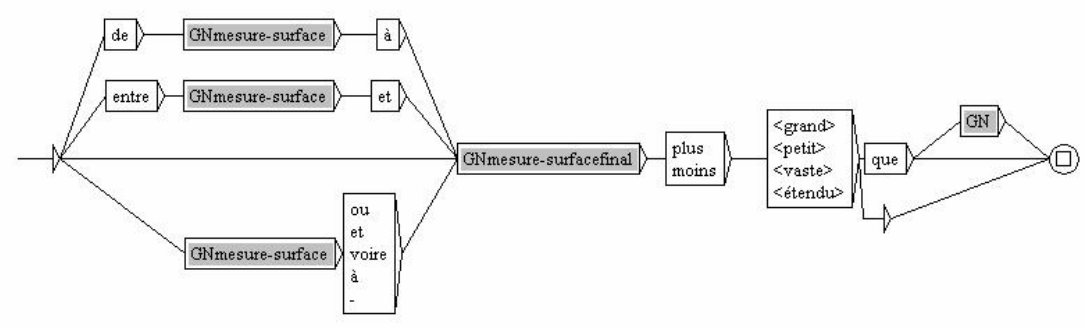

Figure 65 : graphe généré pour *surface*

3.5 Application à des textes

3.5.1 Généralités

L'étude linguistique que nous avons réalisée a pour but ultime de confronter à des textes les contraintes locales codées sous la forme de graphes. Nous disposons d'un ensemble de grammaires important. Pour chaque type de phrase étudiée, nous associons un ensemble de grammaires composé de :

- la grammaire décrivant la phrase de base et l'ensemble de ses transformées (souvent peu fréquentes dans les textes),
- la grammaire décrivant les réductions générées à partir de la phrase de base, c'est-à-dire des groupes nominaux,
- les grammaires représentant des expressions dérivées et partielles de la phrase (groupes nominaux, adverbes, prépositions, déterminants, modifieurs).

Ces grammaires sont construites en général à l'aide de graphes paramétrés et de la méthode d'E. Roche (1993). Nous synthétisons ci-dessous l'ensemble des grammaires construites:

- *Det Ng être de Dnum Unité*
    - phrase : *cette longueur est supérieure à 14 miles*
    - *GN* : *une longueur époustouflante de 130 m = 130 m de longueur*

- *N0 avoir un Ng de Dnum Unité*
    - phrase : *Le camion a un poids de dix tonnes*
    - *GN* : *Le camion d'un poids de dix tonnes*
    - déterminants composés : *10 tonnes de (plutonium)*
    - modifieurs : *haut de 120 m* (adjectival) ; *d'une hauteur de 120 m* (nominal)

- *N0 Vsup Prep un Ng de Dnum Unité Prep1 N1*
    - phrase : *Paul est à une distance de trente kilomètres de Paris*
    - GN : *la distance de trente kilomètres entre Paul et Paris*
    - adverbes : *à une distance de trente kilomètres de Paris*
    - prépositions : *à une distance de trente kilomètres de*

- *N0 être Dnum Unité Adj que N1*
    - phrase : *Paul est 10 kg plus lourd que Max*
    - modifieur : *10 kg plus lourd (que Max + E)*

Avant toute chose, il est nécessaire d'évaluer les grammaires obtenues et leur intérêt pour l'analyse automatique de textes. Nous illustrons ensuite comment elles peuvent être utilisées.

3.5.2 Evaluation des grammaires

Nous souhaitons évaluer nos grammaires en répondant à deux questions :

- La construction et la mise à jour des grammaires sont-elles faciles à mettre en oeuvre ? (production et maintenance)
- Les grammaires sont-elles à la fois complètes et précises ? (rappel et précision)

Ses questions sont légitimes pour pouvoir prétendre utiliser ces données pour l'analyse automatique des textes.

3.5.2.1 Production et maintenance

Nous avons montré dans ce chapitre le processus complet de construction de grammaires d'expressions de mesure. A première vue, la production de telles grammaires n'est pas difficile. En effet, les formalismes utilisés sont extrêmement simples et visuels (tables et automates) et donc facilement compréhensibles pour les linguistes. Malgré cette simplicité apparente, notre méthode basée sur M. Gross (1975) permet de décrire systématiquement des phénomènes très précis. La grosse difficulté réside dans la quantité astronomique de données à accumuler. Ainsi, notre processus requiert une extrême rigueur dans l'organisation des données qui peuvent rapidement devenir illisibles et donc incompréhensibles.

Nous avons vu que nous disposions de deux méthodes de production des grammaires à partir d'une analyse linguistique détaillée :

- par construction manuelle (ex : les graphes **DnumUniteDe** et **ADnumMetreDeN1**)
- au moyen d'une représentation intermédiaire (tables syntaxiques) et d'un mécanisme semi-automatique de conversion en graphes

Ces deux méthodes se mélangent : les graphes patrons utilisent des sous-graphes faits entièrement à la main comme les déterminants numériques et les unités. Il n'existe pas de critères clairs pour choisir l'une ou l'autre. Il faut simplement prendre la méthode la moins contraignante et la plus flexible. La méthode par tables syntaxiques est extrêmement intéressante si la description des expressions nécessite de nombreuses duplications de morceaux de graphes. Les séquences relativement simples (type mots composés) sont très naturellement décrites à la main dans des graphes : par exemple, les déterminants numériques. La construction des graphes *GNmesure*, bien que simple, nécessite la duplication systématique de morceaux de graphes pour chaque classe d'unités. Il est donc préférable d'automatiser ces opérations à l'aide de la méthode d'E. Roche. Pour l'ensemble des unités de mesure, nous aurions pu automatiser une grande partie des opérations de construction car chaque unité a le même ensemble de préfixes (*milli-*, *centi-*, *déci-*, *déca-*, *hecto-*, *kilo-*, etc.). Cependant, les duplications, dans ce cas-là, ne sont pas très contraignantes. Pour les structures de phrase qui sélectionnent un ensemble de prédicats, il est très souvent préférable de construire des tables syntaxiques. En effet, chaque prédicat rentre dans un certain nombre de structures appartenant à un sur-ensemble commun. Cependant, leur comportement diffère dans le détail, ce qui est très difficile à coder manuellement sous la forme de graphes. Il est plus facile de coder les informations syntaxiques dans une table à l'aide de valeurs booléennes. Par contre, si le nombre d'entrées est très réduit, il est peut-être préférable de le faire directement sous la forme de graphes car la gestion des variables dans les graphes patrons n'est pas toujours facile. Notons que les structures *N*0 *Vsup Prep un Ng de Dnum Unité Prep*1 *N*1 sont codées dans une table bien que le nombre d'entrées soit faible, du fait de la répétitivité des duplications.

Le gros désavantage de ces deux méthodes est que le codage des comportements syntaxiques est manuel, donc très coûteux en temps. Cependant, cette approche est nécessaire pour décrire

des phénomènes très précis car chaque entrée lexicale a un comportement propre qui ne peut être prédit automatiquement. Cela n'empêche pas l'utilisation de certains processus automatiques qui atténuent cet inconvénient. Ils permettent d'extraire rapidement des informations linguistiques de grands corpus. Par exemple, il est intéressant de chercher tous les noms prédicatifs *Ng* dans les textes afin d'examiner les contextes droits et gauches et, ainsi, de compléter nos grammaires par les séquences trouvées dans le texte mais manquantes dans les grammaires (exemple : *5 mètres en largeur = une largeur de 5 mètres*). Une fois les unités simples décrites à l'aide de dictionnaires de manière quasi-exhaustive, il est possible de les chercher dans les textes et ainsi trouver des améliorations à la description des déterminants et prédéterminants numériques[48]. Au cours de telles opérations de maintenance, l'utilisation de tables syntaxiques comme représentation intermédiaire permet d'éviter certaines duplications de graphes à la main comme nous l'avons montré ci-dessus. L'application des graphes *GNmesure* permet de compléter la liste des *Ng* (exemple : *âge*, etc.).
D'autre part, notre formalisme n'impose pas de coût supplémentaire en temps lors de la mise à jour des données. En effet, dans les graphes, l'insertion d'une nouvelle séquence linguistique est une opération très simple : l'ajout de nouvelles transitions dans le graphe. La modification des tables est également très simple et très économique. Par exemple, l'ajout d'une nouvelle propriété ne requiert que l'ajout d'une colonne dans la table, la modification du graphe patron et la génération automatique des graphes associés à chaque entrée lexicale.

3.5.2.2 Bruit et silence

Traditionnellement, pour évaluer l'efficacité des grammaires, on effectue des évaluations quantitatives en les appliquant[49] sur un corpus de taille moyenne que l'on décortique ensuite manuellement. Deux critères d'évaluation sont alors calculés : le silence et le bruit. Le silence est la quantité d'expressions pertinentes non trouvées par la grammaire. Le bruit est la quantité d'occurences reconnues par la grammaire mais qui sont incorrectes. En général, ces deux critères sont donnés sous la forme de pourcentages de ces quantités par rapport au nombre d'occurences correctes trouvées manuellement.
Avant de réaliser ce type d'évaluation, nous faisons quelques remarques. Tout d'abord, d'un point de vue général, notre approche purement linguistique se distingue de la plupart des méthodes utilisées en TAL qui, dans la majorité des cas, utilisent des approches statistiques passant par des phases d'apprentissage sur des corpus. Par définition, les résultats sont des approximations et donnent invariablement des erreurs. Le but est d'évaluer quantitativement le modèle statistique utilisé. Par notre méthode, nous décrivons tous les cas possibles et non pas seulement ceux qui apparaissent dans les corpus. Ainsi, les expressions retrouvées dans les corpus ne représentent qu'une infime proportion des expressions réellement représentées dans les grammaires. Ainsi, une évaluation ne porte que sur une toute petite partie de la grammaire. Cependant, ces évaluations permettent de compléter nos descriptions au fur et à mesure au moyen des expressions non trouvées car une grammaire est toujours en évolution. Une évaluation n'est donc valable qu'à un instant *t*. Dans notre étude spécifique, nous constatons que les expressions de mesure dans les corpus journalistiques généraux ou dans les corpus scientifiques de vulgarisation, n'apparaissent que très rarement et leur fréquence selon les types d'unités est très hétérogène. En effet, alors que le mot *kilomètre(s)* apparaît 2 645 fois dans une année du Monde (1994) (environ 100 millions de mots), il n'existe que 19 séquences appartenant aux classes **Volt** et **Volt_abr**. Les unités *Newton* et *électron-volt* n'apparaissent même pas ; les unités mesurant la température n'apparaissent quant à elles pas

[48] L'ambiguïté des symboles des unités nécessite un filtrage manuel important (ex : *a* pour *are* ou *s* pour *seconde*).
[49] Nous appliquons nos grammaires sur un texte prétraité à l'aide du logiciel Unitex avec la règle du 'longest-match'.

plus d'une centaine de fois. Les unités que l'on retrouvent les plus fréquemment sont les unités de mesure de temps (ex : *mois*), de longueur (ex : *mètre*) et de masse (ex : *kilogramme*) et les monnaies (ex : *dollar*). Une évaluation globale est donc difficile. Il faut regarder chaque unité séparément. Cependant, il est clair que les classes d'unités n'apparaissant que quelques fois ne peuvent être évaluées de manière représentative comme pour la famille de *volt*. Nous décidons de nous consacrer aux expressions contenant des unités de longueur et de masse. Les expressions temporelles sont trop ambiguës et très complexes pour pouvoir être correctement traitées par nos grammaires et elles méritent des études approfondies (D. Maurel, 1990 ; M. Gross, 2002). Par ailleurs, nous avons utilisé des groupes nominaux libres extrêmement simples se résumant à l'expression rationnelle *<DET> (<E>+<A>) <N> (<E>+<A>)* car autrement cela amène trop d'erreurs du fait de l'ambiguïté naturelle de la langue. La description de ce type d'expressions étant fondamentale et difficile, nous décidons de ne pas la faire par manque de temps matériel. Ainsi, nous avons des expressions de mesure souvent « tronquées » mais contenant beaucoup d'information : des groupes nominaux lexicalisés (*une altitude de 10 000 m = 10 000 m d'altitude*) ; des modifieurs (*d'une longueur de dix mètres* ; *supérieurs à 30 kg*) ; des prépositions locatives (*à 1 km au nord de*) et des déterminants composés (*trente litres de*). Les unités de mesure que nous évaluons ne sont pas extrêmement fréquentes : il existe seulement 1 240 unités de mesure de longueur sur les vingt premiers millions de mots du corpus. Pour les unités massiques, c'est encore plus difficile à évaluer du fait de l'ambiguïté de certains symboles avec des mots extrêmement fréquents dans la langue, comme *t* ambigu avec le *t* de *Max a-t-il mangé ?; livre* est aussi ambigu avec la monnaie *livre sterling* et l'objet que l'on lit. Si l'on applique la grammaire décrivant les unités massiques, seules 142 occurrences sur 2 000 appartiennent à des expressions de mesure de masse (7%), ce qui n'est pas suffisant pour faire un calcul représentatif. Pour les expressions mettant en jeu des unités de mesure de longueur, notre démarche a été la suivante : nous avons appliqué en même temps nos grammaires représentant des expressions de mesure de longueur avec celles des unités de mesure de longueur. Nous avons arrêté l'application après 2 000 occurrences trouvées. Après examen des occurrences, nous n'avons gardé que 1 240 d'entre elles car certaines unités sont ambiguës comme *m* ambigu avec le *m* de *Max m'a dit que ça allait* et d'autres appartiennent à d'autres types de mesure telles que les mesures de surface ou de volume (*un volume de 10 m3*). Après examen des 1 240 occurrences pertinentes, nous obtenons les résultats suivants :

Silence pur	*Reconnaissance partielle*	*Bruit pur*
7	56	9

La colonne « silence pur » indique le nombre d'expressions de mesure de longueur qui n'ont pas été trouvées automatiquement (même partiellement). La colonne « reconnaissance partielle » donne le nombre de séquences qui n'ont été reconnues que partiellement. La colonne « bruit pur » indique le nombre d'occurrences qui n'auraient pas dû être reconnue du tout. Nous calculons le taux de silence de deux manières : soit sans tenir compte des séquences reconnues partiellement ; soit en en tenant compte. On agit de la même manière pour le taux de bruit.

Taux de silence = 0.6% (5,1%)[50]
Taux de bruit = 0.7% (5,3%)

[50] Le premier résultat est calculé en considérant les occurrences partielles comme correctes alors que le résultat entre parenthèses est calculé en les considérant comme incorrectes.

Pour la plupart des séquences qui n'apparaissent pas, cela est dû à de simples oublis dans la construction des graphes, comme *soixante* qui, malencontreusement, a été oublié dans le graphe des déterminants numériques écrits en lettres.

e à Istrana, une base située à soixante kilomètres à le nord-est de Vicence (Italie), de

La présence d'un adjectif entre le déterminant numérique et l'unité est également une source de silence comme :

toute sa longueur actuelle un petit kilomètre_, que vingt-huit ans après sa naissance

Les reconnaissances partielles sont d'abord dues à des expressions auxquelles nous n'avions pas pensé dans un premier temps comme

J'enfonce dans le terrain deux pieux à vingt et un pieds environ l'un de l'autre.{S} Je

Certaines expressions numériques peu courantes nécessitent une conversion en une mesure avec une unité plus courante pour qu'elle puisse être compréhensible pour le lecteur. Cette conversion peut être insérée en plein milieu de l'expression :

à 5 milles nautiques (environ 9 kilomètres) à le sud de Brest

Plusieurs occurrences partiellement reconnues sont les conséquences d'oublis dans la description des prépositions locatives *Loc* (cf. les deux premières phrases ci-dessous) ou d'erreurs de codage dans les tables (cf. la dernière phrase) :

heures et son rayon de action à 40 km autour de l'hôtel de Manhattan où se tenait la
enfin l'aménagement, à 15 mètres sous terre, de la station Tolbiac-Masséna de
le grand circuit, ce est-à-dire 40 kilomètres de périmètre de temples entretenus.{S}

Notons que nous n'avons pas essayé de reconnaître les adverbes lorsque la séquence *Loc N1* a été transformée en adverbe. Il est prévu d'ajouter ce type de données ultérieurement.

passe la piste qui mène à Tadjourah, 12 kilomètres plus à l'ouest.{S} Les

Les erreurs proviennent parfois du corpus lui-même qui contient des fautes d'orthographes :

aurait coulé dans le lac par 160 m de fonds le 24 janvier dernier après un amerrissa

Dans une revue scientifique telle que *Science et Vie*, la distribution entre les unités est plus homogène que dans *Le Monde*, même si les expressions temporelles sont toujours largement majoritaires. Nous décidons de réaliser une évaluation quantitative globale sur ce type de corpus. Nous avons assemblé un ensemble d'articles de *Science et Vie* datant de l'année 1992 pour former notre corpus (environ 100 000 mots). Nous avons dû faire quelques modifications dans nos grammaires : par exemple, nous avons supprimé la règle du blanc intercalé tous les trois chiffres dans les *DnumEnChiffres* car elle n'était respectée que partiellement par les journalistes.

Nombre total d'occurrences	334
Silence pur	4
Bruit pur	21
Reconnaissance partielle	12

Taux de silence : 1,2% (4,8%)
Taux de bruit : 6,3 % (9,9%)

Les 4 expressions non reconnues (silence) sont dues à trois facteurs différents :

- Des fautes typographiques sont présentes dans le corpus

approchant celle de la lumière:{S} 300 0000 km/s. {S}La masse est une forme de

- Un cas de déterminant nominal n'a pas été répertorié (*le dixième de*) :

dépassant à peine le dixième de mm en longueur pour une épaisseur de 4 ðm. {S}Avec

- certaines unités dans nos grammaires ne se trouvent pas dans le dictionnaire électronique (ex : *méga-électrons-volts*)

Certaines expressions reconnues sont du bruit pur du fait de :

- l'ambiguïté naturelle de la langue et d'une analyse trop locale

un indice de cétane amélioré.{S} De 50 à 52 à l'heure actuelle, ils ne désespèrent pas
s de la peau).{S} L'activation de le VIH-1 a ainsi été reproduite chez un animal entier,
la GT 31 s 'apparente

- certaines expressions n'ont pas été répertoriées

à 10 m près
de la couche de ozone constatée entre 30ø et 64ø de latitude nord entre 1969 et 1986 ne

Les résultats obtenus sont tout à fait satisfaisants. Cependant, cette évaluation n'est valable que pour un instant *t* et un corpus bien précis. Nos grammaires sont mises à jour en permanence : les oublis et les petites erreurs de codage disparaissent au fur et à mesure. Ainsi, nos taux de silence tendent de plus en plus vers 0%. le bruit n'est dû qu'à l'ambiguïté naturelle de la langue et ne peut être supprimé qu'au prix d'une analyse plus contextuelle.

3.5.3 Opérations utilisant les grammaires

Certaines opérations que nous proposons ci-dessous sont pour l'instant théoriques car elles nécessitent une grammaire complète et précise des groupes nominaux.

3.5.3.1 Localisation de constituants syntaxiques

La seule opération directement utilisable à ce jour est la localisation automatique de constituants syntaxiques tels que :

- des groupes nominaux :

  une longueur de 10 m
  10 m de long

- des groupes adjectivaux :

  âgé de 10 ans
  distant de 10 m

- des groupes prépositionnels (modifieurs)

  (un camion) *d'un poids de dix tonnes*

- des prépositions locatives composées :

  deux cents mètres en amont de (la station)

- des adverbes :

  à 10 m de hauteur

- des déterminants nominaux :

  trois tonnes de (pétrole)

Pour réaliser une telle opération, il faut ajouter des informations de sortie aux graphes. Dans le cas où nous gardons le format du DELAF et du DELACF[51], nous obtenons le graphe ci-dessous pour les déterminants nominaux de mesure. Si l'on applique ce graphe à la phrase *Max a mangé 200 grammes de frites*, la séquence *200 grammes de ,.DET+Dnom+Mesure* est générée en mode fusion[52] et peut donc être rajoutée au dictionnaire du texte car elle est compatible avec le format.

[51] Rappel : le DELAF et le DELACF sont les dictionnaires électroniques que nous utilisons.
[52] Dans le mode fusion, les informations de sortie sont insérées dans la séquence reconnue par le graphe.

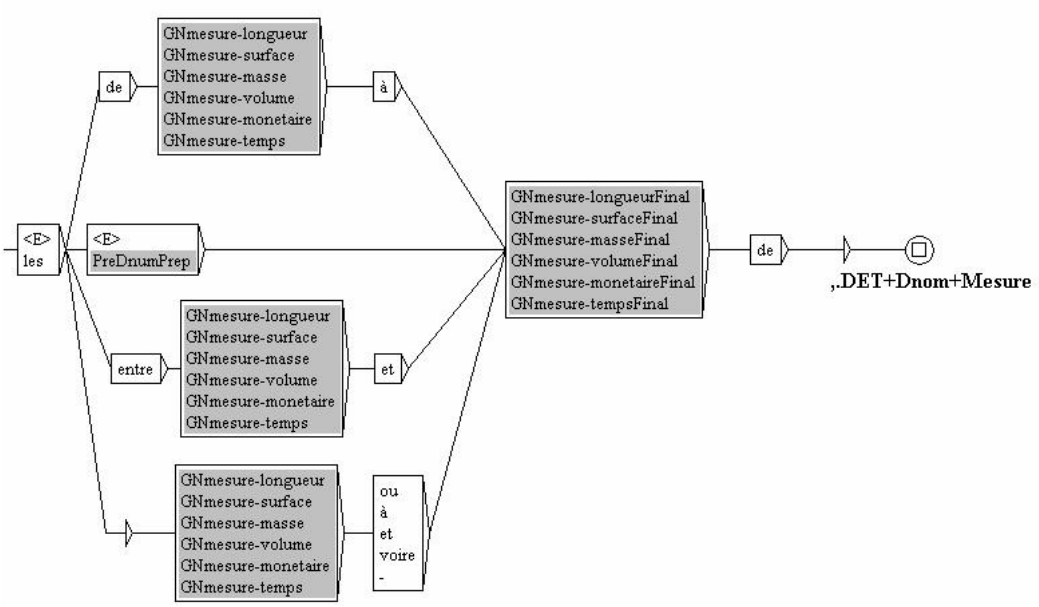

Figure 66 : localisation de déterminants nominaux de mesure

3.5.3.2 Analyse transformationnelle

A plus long terme, nous pourrions utiliser les techniques d'E. Roche pour une analyse transformationnelle d'expressions de mesure à l'aide de transducteurs à variables. En effet, le groupe nominal *une salle à 10°C* peut s'analyser par la séquence suivante :

une salle à 10°C, une salle avoir une température de 10°C.GN+mesure

Cette séquence signifie que l'expression reconnue *une salle à 10 °C* est un groupe nominal qui est le résultat de la réduction de la phrase élémentaire : *la salle a une température de 10°C*. Le graphe utilisé pourrait être le suivant :

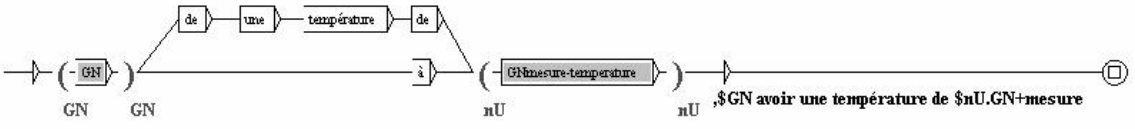

Figure 67 : analyse transformationnelle de groupes nominaux de mesure

Les séquences *$GN* et *$nU* sont des variables. Lors de l'application de ce graphe à la séquence ci-dessus : chaque partie reconnue par les chemins entre parenthèses du graphe est stockée dans la variable associée. Ainsi, *une salle* est placé dans *$GN* et *10°C* est placé dans *$nU*. En sortie de l'application de ce graphe, les variables sont remplacées par leur contenu.

3.5.3.3 Extraction et normalisation d'information

A l'aide des grammaires construites, nous pouvons réaliser des normalisations. En effet, il a été montré à plusieurs reprises (A. Chrobot, 2000 ; L. Karttunen, 2003) que les déterminants numériques cardinaux en toutes lettres pouvaient très facilement être normalisés sous la forme de nombres écrits en chiffres à l'aide de transducteurs, facilitant ainsi le travail de traduction de ces séquences :

français ↔ formel ↔ anglais

dix-sept ↔ 17 ↔ *seventeen*

Cependant, nous avons vu que la syntaxe des nombres en chiffres pouvait dépendre de la langue :

Deux mille trois cent douze ↔ 2 312
Two thousand three hundred and twelve ↔ 2,312

Mais, cela n'est pas un problème si l'on utilise une représentation numérique indépendante de la langue.

Nous pourrions étendre cette application aux mesures. En effet, il existe un symbole standard international, pour chaque unité écrite en lettres. Ainsi, on a :

français ↔ formel ↔ anglais
mètre ↔ *m* ↔ *meter*

Le formalisme du transducteur à états finis est parfaitement adapté. Il suffit donc de reprendre nos grammaires d'unités et leur ajouter une sortie comme ci-dessous avec le graphe **Metre** :

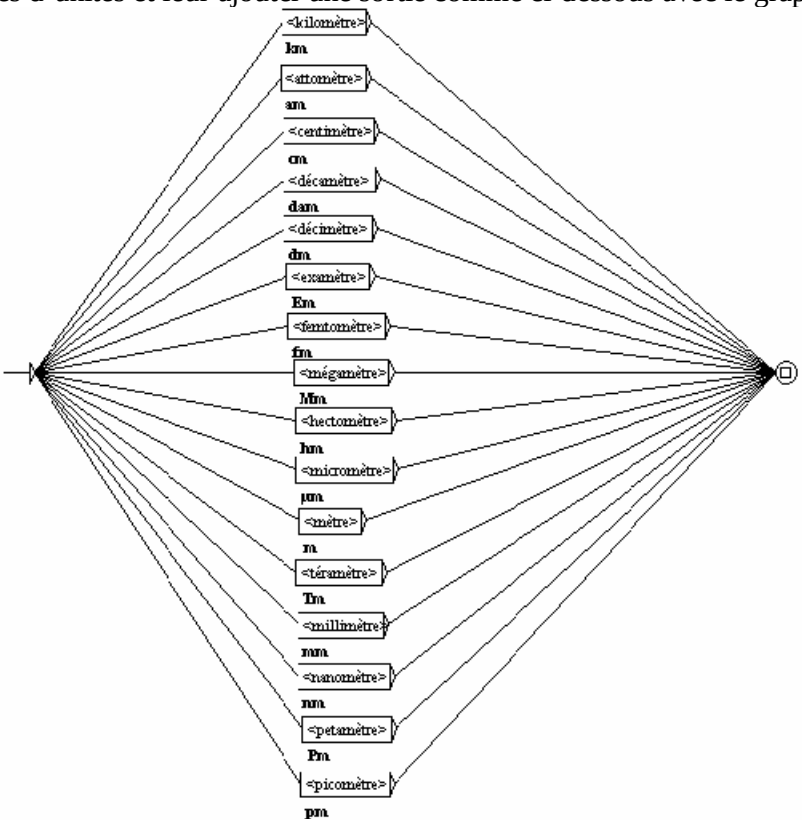

Figure 68 : normalisation du graphe Metre

Nous pourrions même aller plus loin dans la normalisation en convertissant chaque unité (ex : *kilomètre*) en son unité de base (ex : *mètre*). On aurait une conversion comme suit :

kilomètre ↔ *1 000 m*

Il est alors très facile de combiner les normalisations des déterminants numériques et des unités pour normaliser les phrases de mesure (ou leurs réductions) de la manière suivante :

cette salle à 17°C ↔ T(*cette salle*) = 17°C[53]
Paul est à 18 km de Paris ↔ d(*Paul,Paris*) = 18 km[54]

Les transducteurs à variables sont extrêmement efficaces comme le montre le graphe théorique ci-dessous. Lors de l'application de ce graphe à *Paul est à 18 km de Paris*, la séquence reconnue par *N*0 (*Paul*) est stockée dans la variable *$0*, *18km* est stockée dans *$nU* et *Paris* est stockée dans *$1*. Ainsi, en sortie, on obtient en remplaçant les variables par leur contenu : d("*Paul*", "*Paris*") = 18km.

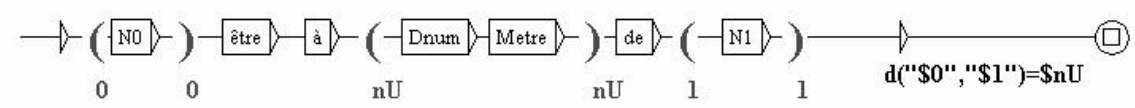

Figure 69 : normalisation

Cependant, des variantes lexicales introduisent des modifications sémantiques. C'est le cas pour :

la longueur de la corde est inférieure à dix mètres ↔ longueur(*la corde*) < 10 m
une longueur de corde d'à peu près dix mètres ↔ longueur(*la corde*) ≅ 10 m

Par ailleurs, il existe des expressions de mesure sous la forme d'intervalles comme nous l'avons montré précédemment avec les structures *entre … et …* et *de … à …*. Là aussi, il y a moyen d'utiliser une normalisation mathématique :

Le spectacle a une durée comprise entre 45 min et 1 h ↔ durée(spectacle) = [45 min, 1 h]

Nous n'entrons pas dans les détails, mais il est clair que cette application mériterait une étude complète.

3.6 conclusion

Nous avons procédé à la description systématique de certaines expressions de mesure sous la forme de grammaires locales. Par cette étude, notre but principal était d'exposer le processus complet de représentation d'un phénomène linguistique relativement simple mais peu étudié jusque-là, en mélangeant deux méthodes différentes mais complémentaires : soit une construction directe en graphes, soit une construction par l'intermédiaire de tables syntaxiques. Nous avons également montré l'intérêt d'une telle étude pour le TAL. Clairement, nous n'avons pas examiné tous les types d'expressions de mesure, notamment les phrases décrivant une évolution où existent des contraintes lexicales fortes comme dans :

Paul a augmenté son poids de 10 kg par rapport à il y a deux mois
Paul a pris 10 kg
*Paul a (grossi + maigri + *grandi) de 10 kg*

[53] T est le symbole de température.
[54] d est le symbole de distance.

Par ailleurs, nous ne sommes pas allé très loin dans l'examen des expressions figées ou semi-figées contenant une mesure. Nous nous sommes simplement contenté du cas *vent de 45 nœuds*.

Chapitre 4 : Analyse et représentation d'adverbes locatifs

4.1 Introduction

L'analyse automatique des groupes prépositionnels (*Prep GN*) dans les textes est un problème bien connu et difficile du domaine du TAL. L'une des principales difficultés consiste à distinguer les groupes prépositionnels arguments des compléments circonstanciels (ou adverbes) :

Le gouvernement donne une subvention à Rome (à Rome = complément essentiel)
Luc se repose (à Rome = complément circonstanciel)

Les principales méthodes de résolution sont statistiques (D. Hindle et M. Rooth, 1994 ; E. Brill et P. Resnik, 1994 ; M. Collins et J. Brooks, 1995 ; J. Zavrel et al, 1997 ; etc.). Il existe des méthodes utilisant des indices linguistiques (C. Fabre et al, 2002). Une autre difficulté consiste à repérer la classe sémantique à laquelle ils appartiennent (temps, lieu, manière, etc.). Les chercheurs lexique-grammairiens ont montré que certaines classes de compléments circonstanciels sont facilement représentables à l'aide d'automates finis lexicalisés comme les dates et les durées (D. Maurel,1990 ; M. Gross, 2002 pour le français ; J. Baptista, 2002, 2003 pour le portugais). Dans ce chapitre, nous décrivons un type particulier de compléments prépositionnels : les compléments locatifs en français. Il existe déjà de nombreux travaux linguistiques généraux sur les constructions locatives (C. Vandeloise, 1986 ; A. Borillo, 1998). Dans le cadre du lexique-grammaire, nous citons A. Guillet et C. Leclère (1992) et J.P. Boons (1985). La plupart des travaux de TAL réalisés sur le sujet cherchent avant tout à décrire des contraintes sémantiques dans ces compléments. Par exemple, certains cherchent à construire des modèles géométriques dans l'espace utilisant notamment des déplacements élémentaires pour décrire les mouvements (Y. Mathet, 2002). Ce sujet est sensible dans le domaine du TAL et quelques projets ont été mis en place pour traiter ces objets linguistiques, en particulier le projet GeoSem (laboratoires GREYC, ESO, ERSS et MEDIA/EPFL) dont une des composantes consiste à repérer des séquences locatives géographiques et à leur assigner un marquage sémantique fin.

La forme générale d'un complément locatif peut être décrite comme la forme nominale prépositionnelle *Loc GN* où *Loc* correspond à une préposition locative et *GN* à un groupe nominal. Notre objectif est de trouver un certain nombre de contraintes locales dépendant du lexique permettant d'améliorer la description de tels compléments locatifs. Nous considérons que nos compléments rentrent dans la construction à verbe support *N*0 *Vsup Loc Det N Modif*. Des études sur les structures *être Prep X* (L. Danlos, 1980 ; M. Gross, 1996) ont montré les fortes contraintes qui existent entre les différents constituants. Nous décidons d'examiner un ensemble limité de noms *N* et particulièrement de séquences nominales formées de noms propres de lieux géographiques et/ou de leurs classifieurs locatifs associés (*Paris* a pour classifieur locatif *ville*). Nous montrerons qu'il existe un certain nombre de contraintes qui peuvent être décrites dans des graphes ou des tables syntaxiques (ensuite transformées en graphes).
Nous ferons d'abord quelques rappels sur les adverbes et les groupes prépositionnels locatifs afin de rendre notre argumentation plus claire. Nous étudierons également les prépositions locatives simples et composées dont nous construirons des grammaires locales. L'application de ces grammaires pointant clairement l'ambiguïté naturelle générée par la reconnaissance locale de telles structures, nous nous consacrons à la description de contraintes locales entre les constituants d'un groupe prépositionnel ayant pour nom tête un nom propre (simple ou composé) de lieu géographique. Dans un premier temps, nous regardons le comportement du couple (*Npr*, *Nc*) dans un groupe nominal où *Npr* est un nom d'un lieu géographique (ex : *Pas-de-Calais*) et *Nc* est son classifieur locatif associé (ex : *région*). Ce couple forme un nom propre composé *Nprc* :

pic du Midi (*Npr* =: Midi ; *Nc* =: *pic*)
mer Méditerranée (*Npr* =: *Méditerranée* ; Nc =: *mer*)
région Pas-de-Calais (*Npr* =: *Pas-de-Calais* ; *Nc* =: *région*)
île de Malte (*Npr* =: *Malte* ; *Nc* =: *île*)

Le degré de figement est variable selon ses élements lexicaux. Par exemple, la séquence *mer de Glace* est plus figée que *mer de Norvège* car la première structure désigne un glacier et pas une mer.

Dans le même temps, nous étudions certaines caractéristiques linguistiques des classifieurs et des noms propres utilisés : déterminants, modifieurs, etc. Enfin, nous décrivons la distribution prépositionnelle des groupes prépositionnels locatifs rentrant dans la structure *N*0 *Vsup Loc Det N Modif* lorsque *N* prend trois formes :

- *N* = : *Nprc*
- *N* = : *Npr*
- *N* = : *Nc*

Nos descriptions sous la forme de tables syntaxiques vont aboutir à un système relationnel de tables syntaxiques, non compatible avec la méthode de conversion des tables en grammaires locales d'E. Roche (1993). Pour confronter nos représentations linguistiques à des textes, nous implantons de nouveaux formalismes et algorithmes de conversion.

4.2 Préliminaires linguistiques

4.2.1 Adverbes généralisés

4.2.1.1 Notions d'adverbes généralisés et d'objets

Un adverbe dans la littérature traditionnelle est un mot invariable élémentaire (ex. *hier*) ou dérivé (ex. *doucement*). Dans le cadre du lexique-grammaire, C. Molinier (1990) a réalisé une classification des adverbes en *–ment* au moyen de critères formels. M. Gross (1986) va plus loin et définit la notion d'adverbe généralisé qui est :

- soit un adverbe traditionnel : *ici, couramment*
- soit un groupe nominal prépositionnel où la préposition est parfois zéro : *dans trois jours, sur l'étagère, en train*
- soit une subordonnée composée d'une conjonction de subordination suivie d'une phrase: *dès que la pluie cessera*

Ainsi, il regroupe sous un même terme trois catégories formelles bien distinctes dans la grammaire traditionnelle. L'un de ses arguments est que ces trois formes répondent en général aux questions en *où, quand, comment, pourquoi,* etc. souvent associées aux compléments circonstanciels. Il définit même la structure globale des adverbes par la formule classique d'un groupe nominal prépositionnel :

Prép Dét N Modif

où chaque élément peut être absent ou contracté. Il rappelle qu'un modifieur peut prendre la forme complétive *qu P* et que la conjonction de subordination est souvent de la forme *Conjs = : Prép (E + ce) que*. Désormais, nous adoptons ce principe et abrégeons le terme adverbe généralisé en adverbe.

Un complément essentiel, de même structure globale qu'un adverbe, est un argument essentiel d'un prédicat. Par exemple, le verbe *donner* possède deux compléments essentiels (un objet direct avec la préposition zéro et un datif avec la préposition *à*) :

Max donne sa clé à la gardienne

4.2.1.2 Distinction entre adverbes et objets

La distinction entre adverbe et complément essentiel (ou objet) est un problème important parce que ces deux éléments ont la même structure de surface et que la détermination des arguments des prédicats (compléments essentiels) est une étape fondamentale de l'analyse syntaxique. Il est généralement admis que, dans un complément essentiel, le choix de la préposition dépend en large partie du prédicat et que, dans un adverbe, la préposition dépend avant tout du groupe nominal. Les adverbes et les objets des verbes sont généralement distingués à l'aide de quelques critères traditionnels. D'abord, les compléments essentiels répondent aux questions en *Prep (que + qui + quoi)*, ce qui n'est pas le cas des compléments circonstanciels qui répondent aux questions en *quand, où, comment,* etc. Les adverbes sont mobiles dans la phrase, ce qui est moins vrai avec les objets. M. Gross (1986) montre à l'aide de quelques exemples que ces critères ne sont pas toujours valables. Ils ne sont ni nécessaires ni suffisants. M. Gross est sceptique quant à leur utilisation car cela « fait perdre toute cohérence au domaine complexe des adverbes » (M. Gross, 1986, p. 22). Il faut traiter les adverbes au cas par cas.

4.2.1.3 La portée des adverbes

Il est très souvent possible d'expliciter la relation entre un adverbe *Adv* et une phrase *P* dans laquelle il peut être inséré. Elle s'exprime à l'aide de phrases simples qui mettent en évidence la portée de l'adverbe. Ces constructions utilisent des verbes-supports *Vsup* comme *être*, *avoir lieu*, *se passer*, etc. Il existe deux cas bien distincts :

- Le cas où l'adverbe porte sur la phrase[55]. Dans cette situation, la relation entre *Adv* et *P* peut s'expliciter à l'aide de la construction

  (E + Le fait) que P Vsup Adv

Par exemple, la phrase :

Max boit du champagne régulièrement

peut être interprétée comme :

Max boit du champagne # Que Max boive du champagne se passe régulièrement

Cette construction est souvent considérée comme théorique car stylistiquement difficile. Il est souvent plus naturel d'utiliser le pronom *cela* portant sur *P* :

P # cela Vsup Adv
= : Max boit du champagne # cela se passe régulièrement

La nominalisation de *P* est parfois plus naturelle comme dans

Max arrive lundi prochain
= Max arrive # l'arrivée de Max a lieu lundi prochain

La relation entre *Adv* et *P* n'est pas toujours facile à mettre en évidence, et même pas toujours possible. L'exemple suivant est tiré de M. Gross (1986) :

A l'étonnement de Paul, Max a réussi son examen

La relation entre l'adverbe *à l'étonnement de Paul* et la phrase simple *Max a réussi son examen* s'explicite à l'aide de la phrase ci-dessous où l'adverbe est présent sous la forme d'une phrase à verbe :

Que Max a réussi son examen étonne Paul

Car la structure combinant *Adv* et *P* n'est pas très naturelle :

? Que Max a réussi son examen se passe à l'étonnement de Paul*

- Le cas où l'adverbe porte sur un argument de la phrase

[55] A. Guillet et C. Leclère (1992) préfèrent utiliser le terme « sous-phrase ».

Parfois, un adverbe ne porte pas sur la phrase mais simplement sur un argument N_i de cette phrase. Dans ce cas-là, la relation s'exprime par la construction suivante plus naturelle que la précédente:

N_i Vsup Adv

Par exemple, dans la phrase ci-dessous :

Max a demandé de bonne foi le résultat du match de football
= Max demande le résultat du match de football # Max est de bonne foi

l'adverbe *de bonne foi* apparaît comme un modifieur de *Max* permutable dans la phrase.

Ainsi, pour résumer, la relation entre un *Adv* et une phrase $P = N_0\ V\ Prep_1\ N_1 \ldots Prep_j\ N_j$ peut s'expliciter par la construction :

(Que P + N_i) Vsup Adv

Ces constructions ont été abondamment étudiées dans le cadre du lexique-grammaire : essentiellement lorsqu'il s'agit de formes figées ou semi-figées, non seulement en français (L. Danlos, 1980 ; M. Gross, 1996), mais aussi dans d'autres langues (portugais : E. Ranchhod, 1989).

4.2.2 Les compléments prépositionnels locatifs

4.2.2.1 Notion de complément locatif

Nous rappelons la structure des compléments locatifs :

Prep Det N Modif (*Prep* peut être zéro)

Intuitivement, pour qu'il soit considéré comme locatif, le nom-tête *N* doit dénoter un lieu. Cependant, il existe des cas où *N* est un lieu mais le complément n'est pas locatif comme dans la deuxième phrase des exemples tirés de A. Guillet et C. Leclère (1992) :

Les Gaulois ont envahi Rome (*Rome* est un lieu)
Les Gaulois ont vaincu Rome (*Rome* est pris pour les Romains et donc non locatif)

Un critère formel pour distinguer les compléments locatifs des autres compléments pourrait être le choix de la préposition.En effet, certaines prépositions comme *dans, à, sur, contre, de, par* ont un caractère locatif et leur présence dans un complément pourrait justifier l'utilisation du qualificatif « locatif ». Par exemple, dans les phrases ci-dessous, les compléments commençant par les prépositions citées ci-dessus sont tous locatifs.

Le loup se cache dans la forêt
Max part à Paris
Léa pose le stylo sur la table
Les affiches sont collées sur le mur
Luc revient de la plage
La balle passe par la fenêtre

Cependant, ces prépositions sont ambiguës et ont des emplois non locatifs. Ainsi, leur présence dans un complément ne garantit pas le caractère locatif de celui-ci. Par exemple, dans les phrases suivantes, les compléments utilisés ne sont pas locatifs.

La température de l'eau est dans les vingt degrés celsius
Max m'a rendu visite à l'improviste
Max porte son choix sur Léa
Georges veut faire la guerre contre tout le monde

Par ailleurs, intuitivement, certaines prépositions telles que *avec* sont considérées comme non locatives. Cependant, il existe des cas où, malgré la présence de ces prépositions, les compléments sont locatifs comme dans :

Max range les fourchettes avec les couteaux

Traditionnellement, on associe les questions en *où* et *Prép où* à la notion de lieu. En effet, cette propriété formelle est valable dans un grand nombre de cas :

Où part Max ? à Paris
Où sont collées les affiches ? sur le mur
D'où revient Luc ? de la plage
Par où passe la balle ? par la fenêtre

Il existe cependant des contre-exemples qui rendent ce critère non valable pour tous les compléments locatifs. J. P. Boons (1985) estime que la question en *où* est un critère suffisant, mais pas nécessaire. En effet, les phrases suivantes sont interdites :

** Où revient Luc ? de la plage*

Où Hannibal marche-t-il ? (J.P. Boons, 1985)
? Sur Rome

On peut même se demander si c'est un critère suffisant. M. Gross (1975) avait déjà entamé la discussion en montrant que certaines infinitives dont il n'est pas clair qu'elles soient locatives répondaient à la question en *où* :

Où va Paul ? Acheter du pain

Certains diront que la réponse à cette question n'est pas *acheter du pain*, mais le locatif *à la boulangerie* qui est le lieu sous-entendu par le procès d'aller acheter du pain (C. Leclère, 2002) :

Où va Paul ? A la boulangerie, pour acheter du pain.

La question en *Prep où* est, quant à elle, clairement non suffisante, comme le montre la phrase :

Par où Max va-t-il commencer ? Par le montage des roues.
(où *par le montage des roues* ne dénote pas un lieu)

Pour conclure sur ce sujet, J.P. Boons (1985) estime que l'on ne peut pas couvrir exactement le champ sémantique du lieu à l'aide de critères formels. On peut seulement s'en approcher.

4.2.2.2 Compléments locatifs et verbes supports

Précédemment, nous avons tenté de distinguer adverbes et compléments essentiels. Pour les locatifs, un même complément peut être soit un adverbe soit un argument. Prenons les exemples suivants :

Max chante dans sa chambre
Max va dans sa chambre

En essayant de relier le complément au reste de la phrase, ces deux constructions réagissent différemment :

Que Max chante se passe dans sa chambre
** Que Max aille se passe dans sa chambre*

Bien que la première construction soit d'une acceptabilité difficile, nous l'acceptons, alors que la deuxième est clairement inacceptable. Nous en concluons que, dans la première phrase, *dans sa chambre* est un adverbe. Il est clair que, dans le deuxième exemple, *dans sa chambre* est un complément essentiel et même obligatoire, la phrase suivante étant interdite :

** Max va*

Soit la phrase suivante :

Marie a sauté dans un lac gelé en Suisse

Elle possède deux compléments locatifs (*dans un lac* et *en Suisse*). Nous montrons qu'ils n'ont pas le même rôle dans cette phrase : *en Suisse* joue le rôle d'un adverbe portant sur la phrase élémentaire (c'est le lieu du procès) alors que *dans un lac* est un complément essentiel du verbe *sauter*.

Que Marie (a + ait) sauté dans un lac s'est passé en Suisse
** Que Marie (a + ait) sauté (E+en Suisse) s'est passé dans un lac*

J.P. Boons (1985) et A. Guillet et C. Leclère (1992) montrent que, à l'instar des phrases contenant un adverbe, les phrases à constructions locatives (i.e. dans lesquelles il existe un complément essentiel locatif) peuvent aussi être analysées à l'aide de constructions à verbes-supports. Il est notamment possible d'employer les arguments dans des phrases qui expriment une localisation, avant et/ou après le procès. Le verbe-support neutre *être* est parfaitement adapté dans ce cas-là. Nous utilisons les notations E_i et E_f pour représenter respectivement l'état initial et l'état final d'un procès. Les exemples suivants n'ont pas besoin d'être expliqués.

Max met l'assiette dans l'armoire
E_f *: l'assiette est dans l'armoire*

Max revient de la plage
E_i *: Max est à la plage*

Max va de Paris à Marseille en voiture
E_i : Max est à Paris
E_f : Max est à Marseille

Ainsi, dans l'exemple précédent contenant deux locatifs, on a aussi l'interprétation :

Marie a sauté dans un lac gelé en Suisse
E_f : Marie est dans un lac gelé.

Les mouvements des différents arguments nominaux peuvent aussi être explicités à l'aide de verbes-supports de mouvement comme *aller*, *venir*, *passer*, etc. (il en existe quelques autres).

Max plonge le savon dans l'évier
Procès : *Le savon va dans l'évier*
E_f : le savon est dans l'évier

Marie prend un fruit (de + dans + de dedans) la corbeille
*E_i : le fruit est (*de + dans + dedans) la corbeille*[56]
Procès : *le fruit vient de la corbeille*

Luc jette une balle par la fenêtre (J.P. Boons, 1985)
E_i : la balle est de ce côté-ci de la fenêtre
Procès : *la balle passe par la fenêtre*
E_f : la balle est de ce côté-là de la fenêtre

Notons que J. P. Boons (1985) va plus loin dans l'interprétation sémantique. Il estime qu'il peut exister plusieurs états finaux dans le procès. Dans la phrase suivante,

Les Russes lancent une bombe sur Paris

Le premier état final pourrait être que la bombe atteint sa trajectoire. Il peut être explicité par la phrase :

La bombe est lancée sur Paris

Ce premier état final induit lui-même un deuxième état final :

La bombe est sur Paris

4.2.3 Les prépositions locatives

4.2.3.1 Prépositions simples

L'une des caractéristiques d'un complément locatif est la présence d'une préposition locative avant le groupe nominal[57]. Pour certains linguistes dont D. Le Pesant (2003), la préposition locative est un prédicat. Cette discussion n'est pas pertinente à notre propos.

[56] cf. C. Leclère et A. Guillet, pour plus de détails.
[57] Même si cela n'est pas toujours le cas : *Luc met les couteaux avec les fourchettes* où *avec les fourchettes* est locatif (*Où Luc met-il les couteaux ? avec les fourchettes.*) alors que la préposition *avec* ne l'est pas.

Il est facile de faire une liste exhaustive des prépositions locatives simples : *dans*, *avant*, *devant*, *sur*, *à*, *contre*, *après*, *derrière*, *sous*, etc. Cependant, comme le montre la section précédente, chacune d'elles a des emplois non locatifs. L'ambiguïté avec le temps, par exemple, est récurrente comme le montrent les ensembles de phrases suivants, avec les prépositions *avant*, *dans*, *à* et *après* :

Max s'arrête avant la forêt (lieu)
Max s'arrête avant la nuit (temps)

Dans la ville, la température a augmenté (lieu)
Dans l'après-midi, la température a augmenté (temps)

A Paris, Marie rencontrera du monde (lieu)
A huit heures, Marie rencontrera du monde (temps)

Max et Marie ont disparu après l'étang (lieu)
Max et Marie ont disparu après leur rencontre (temps)

Lorsque la préposition fait partie d'une expression figée ou semi-figée qui peut être décrite sous la forme de grammaire locale, le problème est résolu rapidement : c'est le cas de *dans l'après-midi* et *à huit heures* qui sont reconnues par les grammaires de dates (D. Maurel, 1990 ; M. Gross, 2002). Dans les autres cas, il faut regarder le groupe nominal qui suit la préposition. En général, si le nom-tête de ce groupe nominal est concret, alors on a très souvent affaire à une préposition locative : *dans la forêt*, *après l'étang*, etc. Cependant, ces constatations ne sont que des tendances. Il existe de nombreux contre-exemples. Des noms concrets peuvent être noms-têtes de compléments de temps :

Max partira (après + dans) deux verres
= Max partira après qu'il aura bu deux verres

Mais le comportement du nom *verre* n'est pas exclusif et ce dernier peut très bien appartenir à un complément locatif :

La fourmi a bu dans deux verres

De même, comment déterminer localement la classe à laquelle appartient l'adverbe comprenant le nom *facteur* dans les phrases suivantes :

Max est passé après le facteur = Max est passé après que le facteur soit passé (temps)
? Sur cette image, Max est assis après le facteur [58] (lieu)

Il existe également un problème avec les noms humains collectifs qui désignent à la fois un lieu et un ensemble de personnes :

A la course de voile, Centrale a coupé la ligne avant cette école (temps)
Pour aller à la maison, tu dois tourner avant cette école (lieu)

Les noms prédicatifs désignant un événement se retrouvent plus facilement dans des adverbes de temps : *après leur rencontre*. Cependant, rien n'est moins sûr avec la phrase suivante où

[58] on considère le *facteur* comme un objet inerte.

l'on est incapable de dire si l'on a un adverbe de temps ou un adverbe de lieu (sans doute les deux).

Au concert de Johnny, Marie a rencontré Luc.
Où Marie a-t-elle rencontré Luc ? au concert de Johnny
(= *dans le lieu où Johnny a fait son concert*)
Quand Marie a-t-elle rencontré Luc ? au concert de Johnny

Par ailleurs, pour une même préposition locative, il existe plusieurs emplois locatifs. C'est le cas de la préposition *sur*. L'interprétation sémantique dépend de la nature (aspect, géométrie, etc.) du nom *N* suivant la préposition. (cf. C. Vandeloise, 1986)

Le plateau est sur la table.
= Le plateau est posé sur la table

La branche est sur l'eau
= La branche flotte sur l'eau

L'alpiniste est sur la falaise
= Luc est accroché à la falaise

Luc est actuellement sur Paris
= Luc est actuellement à Paris

On distingue différents sens pour ces quatre phrases. La première phrase indique un contact de surface entre le plateau et la table dans une position horizontale, le plateau étant au-dessus de la table (cf. dessin ci-dessous).

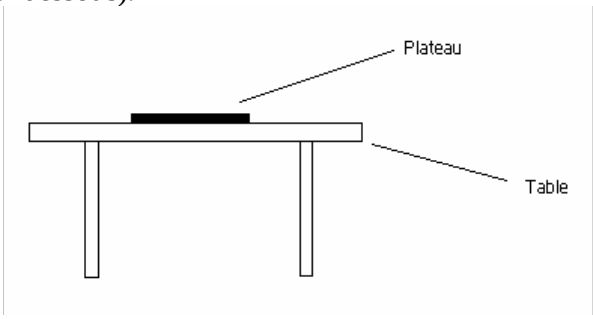

Figure 70 : sur (*plateau*, *table*)

Pour la deuxième phrase, le contact est différent. En effet, la branche est partiellement immergée dans l'eau.

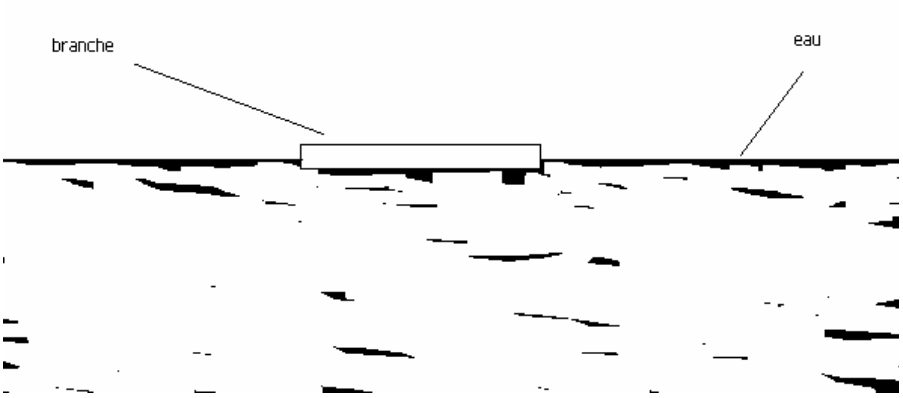

Figure 71 : sur (*branche, eau*)

La troisième phrase indique un contact de surface entre la falaise et l'alpiniste, dans une position verticale (l'alpiniste est accroché).

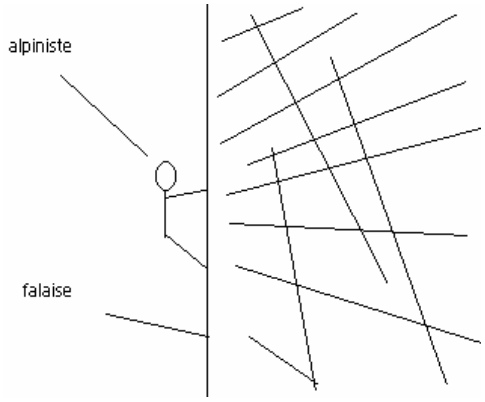

Figure 72 : sur (*alpiniste, falaise*)

En l'absence d'un modèle formel du sens, ces distinctions ne sont pas systématisables. Il existe d'ailleurs des intermédiaires entre ces trois situations.
Enfin, la quatrième phrase est équivalente à *Luc est à Paris*, mais avec une nuance sémantique.

Les prépositions locatives peuvent aussi appartenir à des adverbes figés locatifs (cf. L. Danlos, 1980 ; M. Gross, 1986, 1996). Dans ce cas-là, l'analyse est plus simple car le sens est rattachable à la séquence entière et non représentable par la composition des différents sens des constituants de l'adverbe :

Max est sur la route
Où est Max ? sur la route

Max est à l'asile
Où est Max ? à l'asile

Marie est à l'air du grand large (M. Gross, 1996)
Où est Marie ?A l'air du grand large

En général, les compléments locatifs sont libres et la distribution des prépositions dépend en gros de la géométrie et des propriétés physiques des arguments (cf. C. Vandeloise, 1985).

Cependant, comment peut-on expliquer les différences de distribution des prépositions dans les phrases suivantes où les arguments sont des pièces d'un bâtiment ? (M. Garrigues, 1995).

Luc est (à la + dans la + en) cuisine.
*Luc est (*à la + dans la + *en) chambre*
*Luc est (à la + dans la + *en) salle de bains*
Luc est (à la + dans la + en) salle d'opération

L'utilisation de la préposition *en* peut s'expliquer pour *cuisine* et *salle d'opération* car les phrases impliquent que Luc est l'acteur d'un procès : Luc travaille dans la cuisine et Luc subit ou pratique une opération dans la salle d'opération. Par contre, la distribution de *à* est difficilement explicable. Il en est de même pour les noms *rue* et *place*. Pourquoi observe-t-on une différence de distribution entre ces deux noms pourtant proches ?

*Luc est (dans + *sur) la rue*
*Luc est (*dans + sur) la place*[59]

On peut imaginer que l'utilisation de la préposition *dans* avec le nom *rue* s'expliquerait par le fait qu'une rue peut être vue comme une boîte ouverte, les côtés fermés étant la route et les bâtiments longeant cette route. Il ne semble pas exister d'explication pour *place* (la ressemblance avec un plateau serait une explication bien fantaisiste). Ainsi, il n'est pas évident de prédire exactement la distribution prépositionnelle à partir d'une analyse sémantique basée sur la géométrie et différentes propriétés physiques des arguments. Une solution consisterait à étudier systématiquement le comportement de chaque nom (M. Garrigues, 1995) et de coder pour chacun leur distribution prépositionnelle.

4.2.3.2 Prépositions composées

Nous nous appuyons ici sur les travaux d'A. Borillo (1989) et M. Gross (1996). Nous regardons de plus près les prépositions locatives composées dont la grande majorité ont la forme interne *Prep Det N de* : *à l'intérieur de*, *en amont de*, etc. On travaille ainsi sur la construction *N*0 *Vsup Prep Det N de N*1. A. Borillo (1989) explique que le nom *N* sert à localiser *N*0 par rapport à *N*1. Il ajoute une précision supplémentaire par rapport aux prépositions simples. Il exprime soit une localisation interne soit une localisation externe. Les noms de localisation interne indiquent que *N*0 est localisé au contact ou à l'intérieur de *N*1 : ex. *à l'extrémité de*, *en haut de*, *sur le coin de*, etc. Les noms de localisation externe permettent de localiser *N*0 par rapport à *N*1, mais il n'y a ni contact (*à l'extrémité de la tige*) ni inclusion (*à l'intérieur du combiné*) entre les deux arguments : *à côté de*, *à droite de*, etc. A. Borillo note également la combinaison d'adjectifs de localisation interne (*central*, *supérieur*, etc.) avec des noms qui désignent des partitions de l'objet *N*1 (*partie*, *zone*, etc.) :

La fourmi se trouve sur la zone extérieure du mur
Marie est dans la partie nord de l'île

Elle a systématiquement examiné chaque nom et chaque adjectif, puis codé leurs distributions lexicales dans des tables[60] dont nous nous sommes servi pour construire manuellement des graphes décrivant des prépositions locatives composées. Nous complétons ces listes par celles de M. Gross (1996). Nous prenons notamment les prépositions composées répertoriées dans la

[59] *Luc est dans la place* a un autre sens.
[60] E. Laporte (2002) a aussi réalisé une étude sur ces adjectifs en se servant de cette liste.

table **EPCDN** de M. Gross qui traite des noms (C) figés dans la construction : *N*0 *être Prep C de N*1. Ces expressions ne sont pas uniquement locatives, celles qui le sont sont même minoritaires : environ 36% des entrées ont un emploi locatif (337 sur 933 entrées). L'extraction des expressions locatives est facilitée par le fait qu'une colonne de la table indique si l'expression répond à la question en *où*. Les entrées candidates sont donc celles qui ont un signe '+' à cette colonne. Par ailleurs, nous ajoutons les prépositions ne rentrant pas dans la construction *Prep Det de* comme *face à*. Les graphes construits dans la section sur les expressions de mesure et représentant des prépositions locatives sont également intégrés à l'ensemble des graphes des prépositions composées.
Par ailleurs, une bonne proportion des prépositions composées extraites de ces listes peuvent être transformées en adverbes figés par effacement du groupe nominal libre :

Max est à l'ouest (de Paris + E)
Luc est à (l'intérieur + côté) (de la maison + E)

Certaines n'acceptent pas cette transformation d'effacement :

*Max est en bord (de fleuve + *E)*

Cette propriété a été codée par M. Gross dans la table **EPCDN** dans une colonne. Comme précédemment, nous extrayons manuellement les entrées candidates. Nous complétons nos listes par celles données par D. Le Pesant (2002).

Les prépositions composées permettent de localiser précisément *N*0 par rapport à *N*1, mais cela n'empêche pas que certaines soient ambigues :

Max est au bord du (suicide + fleuve)
*Où est Max ? Au bord du (*suicide + fleuve)*

Cette scène est à la limite du supportable
*Où est cette scène ? *à la limite du supportable*

On note que l'ambiguïté avec les dates est toujours aussi récurrente :

Notre empereur préféré est né à la fin du 18ème siècle. (date)
Luc a épousé Léa au début de l'année dernière. (date)

Nous organisons nos graphes de la manière suivante. Nous représentons 8 formes de prépositions composées :

- *à Det N de*[61] = : *à l'ouest de, à la frontière de*
- *en N de* [62]=: *en amont de*
- *sur Det N de* =: *sur le bord de*
- *dans Det N de* =: *dans le centre de*
- *dans la zone Adj de*[63] = : *dans la zone (ouest + supérieure) de*

[61] Le graphe associé est **ADetNDe**. Pour clarifier le graphe, nous représentons les directions du type *à l'ouest de* dans un graphe différent **ALeOuestDe**.
[62] Le graphe associé est **EnNDe_Loc**.
[63] Le graphe associé est **DansLaZoneAdjDe**. Les adjectifs répertoriés ont des caractéristiques purement géométriques dans l'espace (*supérieur, gauche, extérieur,* etc.). Le sous-graphe **Adj-direction** décrit les adjectifs

- *à Dnum Metre de =: à 10 km (E + à vol d'oiseau) de Paris, à 20 kilomètres au nord de*
- *à un N de Dnum Metre de = : à une hauteur de 30 m de*
- résiduels =: *face à, hors de*

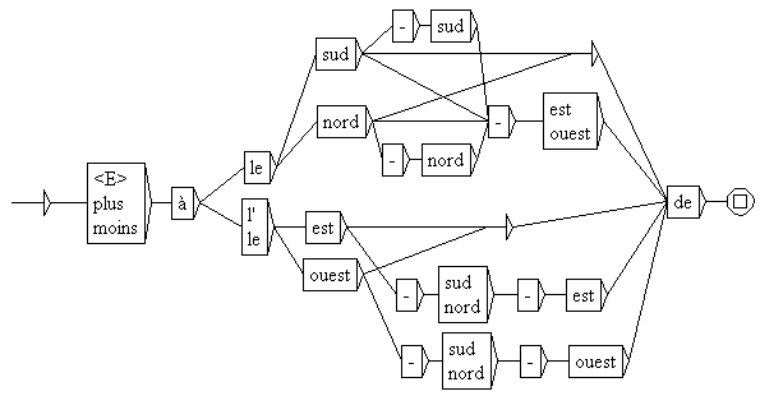

Figure 73 : ALeOuestDe

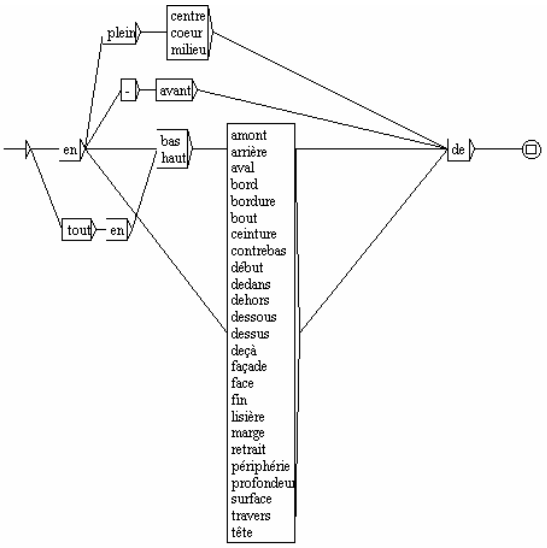

Figure 74 : EnNDe_Loc

nord, est, sud-ouest, etc. Notre grammaire ne reconnaît pas des expressions telles que *dans la zone néerlandaise de la ville* où *néerlandaise* n'est pas une caractéristique spatiale mais une caractéristique démographique de la ville.

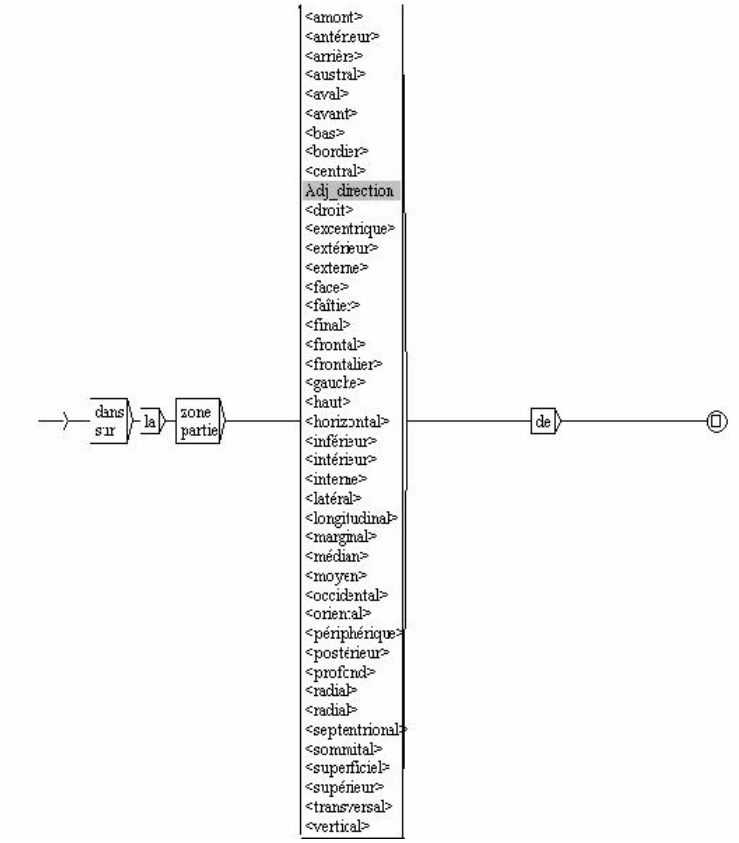

Figure 75 : DansLaZoneAdjDe

4.2.3.3 Quelques statistiques

Nous avons montré dans l'absolu que la localisation d'un complément locatif ne peut se résumer au repérage d'une préposition locative et d'un groupe nominal du fait de l'ambiguïté des prépositions. Dans cette section, nous mesurons quantitativement le taux d'erreur qu'un tel processus introduirait en supposant que la reconnaissance du groupe nominal soit parfaite[64]. Nous prenons une année du journal *Le Monde* (1994) et regardons manuellement le comportement des 1 000 premières prépositions appartenant à l'ensemble {*dans*, *avant*, *devant*, *sur*, *contre*, *après*, *derrière*, *sous*}. Au total, 413 occurrences de ces prépositions ont un emploi locatif, soit 41,3%. La proportion d'emplois locatifs la plus élevée est de 81% avec la préposition *dans*, ce qui très insuffisant. Par ailleurs, nous constatons que certaines prépositions comme *avant* ou *après* appartiennent quasiment toujours à des adverbes de temps (respectivement 90% et 86% des cas). Sur l'ensemble des prépositions trouvées, nous n'avons que 31 prépositions appartenant à des adverbes figés ou semi-figés de temps facilement repérables par des grammaires locales de temps existantes (soit 3,1% de l'ensemble des prépositions) comme

dans l'après-midi
dans la soirée de jeudi
avant le 18 mars

Ceci montre bien la difficulté qu'il y aura à distinguer emplois locatifs et temporels de manière fiable. A terme, le meilleur moyen sera de regarder le groupe nominal du

[64] Ce qui est clairement impossible dans l'état actuel des recherches en linguistique appliquée.

complément locatif. Par exemple, si le nom-tête est un prédicat, la probabilité d'avoir un adverbe de temps sera très grande ; par ailleurs, si c'est un nom concret, la probabilité d'avoir un adverbe locatif sera aussi très grande.

Certaines prépositions comme *contre* ou *sur* sont souvent des prépositions attachées à des prédicats du type

Max proteste contre l'inflation
Luc fait une enquête sur la vie mouvementée de Max

Dans ces cas-là, l'ambiguïté est résolue naturellement par l'analyse syntaxique de la phrase.
Quelques résultats détaillés sont donnés dans la table ci-dessous qui utilise les notations suivantes :

NO : Nombre d'occurrences
NOL : Nombre d'occurrences à emploi locatif
%NOL : pourcentage d'occurrences à emploi locatif

Loc simple	NO	NOL	%NOL
dans	426	345	81
avant	51	0	0
devant	18	9	50
sur	294	76	26
contre	67	0	0
après	96	2	2
sous	45	3	7
derrière	3	2	67
Total	1000	413	41

Table 9 : proportion d'emplois locatifs des prépositions {*dans, avant, devant, sur, contre, après, derrière, sous*} dans notre corpus

Ci-dessous, nous donnons un ensemble d'exemples trouvés dans les textes où la préposition reconnue n'a pas un emploi locatif :

[Noms composés]

après-guerre
après-midi

[adverbes figés ou semi-figés]

avant tout
avant la fin de l'année
dans le même temps
dans les mois à venir
dans ces conditions

[adverbes libres]

dans un hébreu limpide

[prédéterminants]

dans la limite mensuelle de 1 000 francs

[prépositions d'un argument d'un prédicat]

Furieux devant l'entêtement de le jeune douanier (…)
un recours hiérarchique devant le garde de les sceaux contre le juge
(…) a lancé une croisade nationale contre la violence à la télévision

[conjonctions de coordination]

sept à huit heures de avion de Tokyo contre onze de Los Angeles

[conjonctions de subordination figées]

dans la mesure où des enfants meurent, les (…)

[phrases figées]

Sommée de balayer devant sa porte, l'industrie (…)
Max a du pain sur la planche
La jeune indienne a été tuée sur le coup

Parmi les phrases figées données ci-dessus, aucune ne répond à la question en *où* :

*Où l'industrie est-elle sommée de balayer ? *devant sa porte*
*Où Max a-t-il du pain ? * sur la planche*
Où la jeune indienne a-t-elle été tuée ? sur le coup*

Notons que nous avons traité les prépositions *à* et *en* séparément car elles sont très fréquentes par rapport aux autres. Nous avons trouvé 30% de prépositions *à* qui ont un emploi locatif dans le texte (40 % pour *en*)[65]. Pour les emplois locatifs, la très grande majorité de ces prépositions appartiennent à des adverbes contenant un nom propre de lieu (*à Paris*, *en France*). Ainsi, un bon moyen d'améliorer le repérage local des adverbes locatifs est d'étudier le comportement des noms propres de lieu dans ces adverbes (cf. sections suivantes).

D'autre part, nous avons voulu mesurer quantitativement l'apport de la reconnaissance des prépositions locatives composées dans le repérage des compléments locatifs. Nous regardons les prépositions du type *Loc Det N de*. Nous avons appliqué les grammaires de ce type de préposition et examiné manuellement les 1 000 premières occurrences. Nous constatons que 529 d'entre elles sont correctement analysées et ont un emploi locatif (soit environ 53%). Nous donnons ci-dessous quelques exemples d'emplois non locatifs trouvés :

[65] C'est un calcul assez approximatif car nous avons regardé les 200 premières occurrences de chacune des prépositions. Mais notre but est de donner une vague tendance.

[prépositions non locatives]

Luc a été gentil à l'endroit de Marie (*à l'endroit de = envers*)
L'argent est à la base des malheurs de Paul (?à la base du fait que P)
A côté de la crise de septembre, celle-ci est mineure (*à côté du fait que* P)

[adverbes figés non locatifs]

Au bout du compte, Léa est très accueillante

[phrases figées non locatives]

Max est passé à côté de quelque chose de grand

[modifieurs du nom]

L'engagement à droite de cet historien n'est pas passé inaperçu

D'autres prépositions ont un emploi locatif mais sont mal analysées du fait de l'ambiguïté du nom *N* :

Le lieutenant Martin a été affecté à la base de Pau (*N = : base (E + militaire)*)

Dans une perspective d'une analyse sémantique, la reconnaissance des prépositions composées augmente la précision de la localisation car le nom *N* dans ces prépositions indique une information supplémentaire de localisation. La reconnaissance des compléments locatifs avec des prépositions composées est un peu plus facile que celle des compléments locatifs avec des prépositions simples (écart approximatif de 10%). Le comportement des prépositions est très hétérogène. En effet, dans beaucoup de cas, soit toutes les occurrences sont locatives (*à l'ouest de*, *au cœur de*, etc.), soit aucune occurrence (ou presque) n'est locative (*au début de*, *à la fin de*, etc.). En utilisant une approche statistique, ce résultat revient à considérer que les prépositions du deuxième cas (ex : *au début de*) ne peuvent être locatives, ce qui n'est pas vrai dans l'absolu. Par contre, certaines ont un comportement relativement équilibré : *au milieu de* avec 51% de locatifs. Des formules statistiques simples pourraient largement améliorer les résultats grâce aux comportements extrêmes. Cependant, les prépositions composées sont très peu fréquentes par rapport aux prépositions simples comme le montre le tableau ci-dessous. Ainsi, il n'est pas possible d'améliorer les résultats généraux de manière significative en améliorant l'analyse des compléments locatifs à préposition composée.

NO : nombre d'occurrences de *Loc*
NCL : nombre d'occurrences de *Loc* dans une préposition composée locative de la forme *Loc (E+Det) N de*
%NCL1 : pourcentage de prépositions composées locatives (*Loc Det N de*) par rapport au nombre de *Loc* à emploi locatif ou pas dans le texte
%NCL2 : pourcentage de prépositions composées locatives (*Loc Det N de*) par rapport au nombre de *Loc* à emploi locatif dans le texte

Loc simple	NO	NCL	%NCL1	%NCL2[66]
à	3059	56	1,8	6,1
sur	486	2	0,4	1,6
en	1455	4	0,3	0,7
Total	5000			

Table 10 : proportion de prépositions composées par rapport aux prépositions simples

4.3 Grammaires locales de noms propres composés de lieu

4.3.1 Remarques préliminaires

L'objectif principal de ce chapitre est d'étudier le comportement syntaxique d'un ensemble de noms *N* au sein de la construction locative *N*0 *Vsup Loc Det N Modif*. Nous limitons cet ensemble aux noms propres de lieux géographiques :

La fête se passe (à Paris + sur la Seine)
Marie se trouve en Californie
Luc est dans la mer Méditerranée

Cette étude présente beaucoup d'intérêt car elle complète les travaux effectués sur les expressions en *être Prep X*. Elle est également un sérieux apport pour l'analyse automatique de textes du fait de la haute fréquence des noms propres de lieu dans certains types de corpus tels que les textes journalistiques. Avant toute étude sur ces constructions, il est nécessaire de comprendre le comportement interne des noms propres de lieu. Nous proposons une description entièrement lexicale et syntaxique, basée sur la méthodologie du lexique-grammaire.

Des études linguistiques ont été consacrées aux noms propres depuis longtemps comme le montre le volume de *Langages* (J. Molino, 1982) consacré à ce sujet. Mais, depuis quelques années, elles ont pris une nouvelle ampleur du fait du développement du TAL et de la fréquence des noms propres dans les textes. Pour toutes les références sur ce sujet, nous conseillons au lecteur de se référer à K. Jonasson (1995). De manière générale, les études sur ce sujet ont cherché à donner un sens aux noms propres car leur syntaxe paraît limitée. Dans cette étude, nous regardons les noms propres de lieu d'un autre point de vue. En effet, nous considérons ces objets linguistiques comme des formes composées comprenant un classifieur de lieu et nous remettons la syntaxe au centre de la discussion. Par exemple, *Méditerranée* est la forme réduite de *mer Méditerranée* et a un comportement syntaxique différent de *mer du Nord*. Par une étude systématique, nous montrons que la méthodologie du lexique-grammaire est parfaitement adaptée au traitement de ces objets.

Dans le domaine du TAL, les principaux travaux réalisés jusqu'à présent consistent à localiser les noms propres dans les textes et à leur assigner des classes sémantiques : T. Wakao et al. (1996), J. Senellart (1998), A. Cucchiarelli et al. (1999), N. Fourour et al. (2002), N. Friburger (2002), etc. Les méthodes utilisées consistent essentiellement à exploiter la présence de classifieurs : par exemple, un nom propre précédé du nom *Monsieur* (*Monsieur Chirac*) est classé comme un nom de personne. Les systèmes de réponses automatiques à des questions sont une des applications de ces classifications automatiques : O. Ferret et al. (2001), C. Fairon et P. Watrin (2003), etc.

[66] Calculé par la formule suivante %NCL2 = NCL /(%NOL*NO) avec %NOL étant le pourcentage des prépositions simples *Loc* ayant un emploi locatif. (exemple pour *à* : %NCL2 = 56/(30/100*3059).

Du fait que les noms propres en général sont en nombre « quasi-infini » et varient énormément au cours du temps, il est impossible de tous les répertorier dans des dictionnaires, d'où l'intérêt d'outils automatiques d'extraction et de classification. Certaines classes sont un peu plus fermées et plus stables, c'est le cas des toponymes. Ainsi, D. Maurel et O. Piton (1998) se sont lancés dans la construction d'un dictionnaire de toponymes[67] (précisément les noms de régions, de pays et de villes) et d'hydronymes : le dictionnaire électronique *Prolintex*. Ce projet est clairement gigantesque mais son intérêt est indéniable pour le TAL. Chaque entrée du dictionnaire contient plusieurs types d'informations :

- d'ordre linguistique : identification de la classe, information flexionnelle, information syntaxique sur les déterminants pour certains toponymes comme les villes ;
- d'ordre extra-linguistique (positionnement géographique : par exemple, *Paris* est une ville de France)[68]

Par exemple, l'entrée *Tours,.N+PR+DetZ+Toponyme+Ville:ms:fs* (provenant de la deuxième version sans information d'ordre extra-linguistique) signifie que *Tours* a les caractéristiques suivantes : c'est un nom propre (*N+PR*), toponymique (*+Toponyme*), désignant une ville (*+Ville*), qui n'a pas de déterminant obligatoire (*+DetZ*), qui peut s'accorder au masculin singulier et au féminin singulier (*:ms:fs*). Le premier intérêt de ce dictionnaire est de reconnaître avec une grande précision les noms propres de lieu. Ensuite, il est une première base solide pour un système de traduction des toponymes (T. Grass et al., 2002). Les outils d'extraction automatique des noms propres peuvent servir à trouver de nouvelles entrées candidates à ce dictionnaire.

Dans cette partie, nous étudions le comportement syntaxique de noms propres composés notés *Nprc*. En général, les noms propres de lieu que nous utilisons sont des formes réduites (ou courtes) de séquences composées. En effet, chaque lieu a un nom (*Npr*) qui peut être complexe (*Pas-de-Calais, Los Angeles, etc.*) et appartient à une classe de lieu (définie par un ou plusieurs classifieurs *Nc*) : *région Pas-de-Calais, ville de Los Angeles*. Dans certains cas, le nom du lieu est suffisant pour désigner ce lieu comme *Pyrénées-Atlantique* ou *Méditerranée*. Ce sont en fait des formes réduites de noms propres composés comprenant un classifieur *Nc* et un nom *Npr* :

> *Pyrénées-Atlantiques* est la forme courte de *département de les Pyrénées-Atlantiques*
> *Méditerranée* est la forme courte de *mer Méditerranée*

Désormais, nous désignons ces noms propres composés par le couple (*Npr, Nc*) : nous notons la forme longue *Nprc* et la forme courte *Npr*. Notre approche consiste à regarder la composition interne de tels noms composés et à constituer des lexiques en s'appuyant sur les travaux de D. Maurel et O. Piton. Notre travail se distingue de ce dernier par le fait, notamment, que nous ne nous intéressons pas aux propriétés extra-linguistiques et nous codons dans nos lexiques de nouvelles contraintes lexicales et syntaxiques qui foisonnent dans ce type de noms. Ce travail sert de base à notre étude sur la distribution prépositionnelle dans les adverbes locatifs. Dans un premier temps, nous justifions le terme de nom composé utilisé pour le couple (*Npr, Nc*) à l'aide d'arguments linguistiques. Dans ce cas, nous parlerons plutôt de noms propres composés étendus. Ensuite, nous effectuons une classification formelle des noms propres composés de lieu. Puis, nous examinons les contraintes syntaxiques internes auxquels ils sont soumis. Par la suite, nous regardons leur comportement dans des groupes nominaux simples : déterminants et modifieurs. Enfin, nous

[67] cf. aussi D. Maurel et al. (1995).
[68] La deuxième version sortie en 2003 ne contient plus ce genre d'information et se trouve à l'URL suivante : http://www.li.univ-tours.fr/BdTln/Prolintex.html.

codons toutes les contraintes trouvées sous la forme de tables syntaxiques et de grammaires locales. Notons que nous n'avons pas la prétention de couvrir tous les noms propres composés de lieu existants (et loin de là). Il existe trois raisons à cela :

- la liste exhaustive de tous les noms propres de lieu est immense (cf. la construction de *Prolintex* qui dure depuis huit ans).
- le comportement syntaxique est souvent flou pour les noms de lieu peu connus : nous avons consitué nos lexiques à partir de nos connaissances géographiques qui relèvent de la culture générale et nous verrons, malgré tout, que les données accumulées comprennent un grand nombre de contraintes syntaxiques.
- notre objectif est surtout méthodologique : montrer une méthode claire et rigoureuse d'accumulation de noms propres de lieu.

Ce sujet présente une autre difficulté. Un nom propre peut admettre plusieurs classifieurs sans être ambigu pour autant : ainsi, une ville peut aussi être une station de ski.

la (station de ski + ville) de Courchevel
*la (*station de ski + ville) de Paris*

Pour certains noms propres, le classifieur n'est pas clair comme pour les petites villes : est-ce un bourg, un village, une bourgade ? Il existe un autre cas où le classifieur n'est pas clair : les noms de massifs montagneux comme les *Alpes*. En effet, il est possible de dire *(massif + chaîne) des Alpes*. S'il n'existe pas de classifieur clair dans la forme longue d'un nom propre répertorié, nous ne tenons pas compte de ce nom propre. Pour les villes, nous ne traitons que les grosses villes qui sont clairement des villes (ex : *Paris, Bordeaux, le Pirée, La Havane*).

4.3.2 Critères de définition des Nprc

Des critères formels sont nécessaires pour distinguer les différents types de séquences de la forme *Detc Nc de (E + Det) Npr* :

(a) *Une ville de France*
(b) *La ville de Pagnol*
(c) *La ville de Paris*[69]

Nous distinguons formellement ces trois cas en examinant les phrases élémentaires de base à partir desquels sont formés les groupes nominaux :

(a) N0 être situé Loc N1 (*Nc* nom tête de *N*0 et *Npr* celui de *N*1)
UN Nc qui être situé Loc (E + Det) Npr =: *une ville qui est située en France*
= * *UN Nc de Loc (E + Det) Npr* =: *une ville de en France* (M. Gross, 1986)
= *UN Nc de (E + Det) Npr =: une ville de France*

(b) N0 vivre Loc N1 =: Pagnol vit dans une ville
la ville (où vit + de) Pagnol[70]

(c) (E + Det) Npr être UN Nc =: Paris est une ville
la ville (qu'est + E) Paris

[69] On suppose que *Paris* est un nom de ville et non un nom de personne par exemple.
[70] Il existe bien d'autres interprétations pour ce type de groupe nominal.

Dorénavant, la structure que nous étudions est la structure (c).

4.3.3 Statut syntaxique des Nprc

Quel est le statut syntaxique des séquences que nous traitons ? Une première approche consisterait à considérer ces séquences comme des appositions (M. Rothenberg, non daté). Les séquences *la mer Méditerranée, la ville de Paris, l'état de Californie* et *l'île d'Ouessant* peuvent être mises sur le même plan que *le roi Louis XIV* où la séquence *Louis XIV* est traditionnellement considérée comme une apposition. La préposition *de* se trouvant entre *Nc* et *Npr* est le principal obstacle à cette solution même si elle est parfois considérée comme une préposition neutre d'apposition[71]. La séquence *le roi Louis XIV* est à raprocher de la phrase classificatrice réflexive :

Louis XIV est un roi / Ce roi est Louis XIV

ce qui revient à admettre l'hypothèse d'une transformation

le roi qu'est Louis XIV
= le roi Louis XIV

On observe le même comportement pour *la ville de Paris* et *l'île d'Ouessant.*

Paris est une ville/ Cette ville est Paris
Ouessant est une île / Cette île est Ouessant

La ville qu'est Paris = la ville de Paris

La séquence *la mer méditerranée* se comporte différemment alors qu'elle a la même forme de base (sans *de*) que *le roi Louis XIV* : le déterminant *la* précédant *Méditerranée* est obligatoire dans la phrase classificatrice.

*(*E + la) Méditerranée est une mer / Cette mer est (*E + la) Méditerranée*

Il en est de même pour *l'état de Californie* qui a obligatoirement besoin du déterminant *la* dans la phrase classificatrice :

*(*E + la) Californie est un état / Cet état est (*E + la) Californie*

L'utilisation du déterminant indéfini *un* sans modifieur est interdite comme dans les appositions :

*G. Bush n'apprécie pas (le + *un) président Chirac*
*Les français n'appréciaient pas (le + *un) roi Louis XIV*

*Les touristes apprécient (l' + *une) île de Ouessant*
*Luc contemple (la + *une) mer Méditerranée*

[71] Les cas d'alternance entre la forme en *de* et la forme sans *de* sont rarissimes :

la région (E + du) Nord-Pas-de-Calais

Dans les séquences figées telles que *mer du Nord, île du Diable, vallée de la Mort* ou *mer Noire*, les phrases classificatrices dissociant *Nc* et *Npr* ne sont pas autorisées :

**Le Nord est une mer*
**Le Diable est une île*
**La Mort est une vallée*
**(E + La) Noire est une mer*

Ainsi, les groupes nominaux qui nous intéressent présentent à la fois des ressemblances et des différences avec les appositions. Il en est de même de certaines formes à appositions sans *de* : la séquence *le bâtiment A* ne peut être rapprochée d'une phrase classificatrice telle que *A est un bâtiment*.

Dans l'une des dernières conversations que nous avons eue avec lui, M. Gross parlait de déterminant nominal[72] pour la séquence *la ville de*, ce qui pourrait sembler exact à première vue :

Les touristes apprécient (la ville de + E) (Toulouse + New York + La Havane)

La possible insertion d'adjectifs fragilise cette analyse. En effet, les déterminants nominaux acceptent à peu près uniquement des adjectifs intensifs :

Cette entreprise a embauché une (E + belle + grosse) fournée d'ingénieurs
Cette entreprise a embauché une fournée (E+ hallucinante + importante) d'ingénieurs
Cette entreprise a embauché (des + trois cents) ingénieurs

La variation lexicale des modifieurs des déterminants nominaux est bien plus étendue avec *la ville de* :

La ville (surpeuplée + polluée + américaine + …) de Mexico

G. Gross (1991) parle de construction inverse et place les constructions du type *la ville de Paris* sur le même plan qu'une séquence du type *ce salaud de*[73] :

(*Ce + Le) salaud de Luc m'a craché à la figure*

En effet, il montre que les séquences *ce salaud de* et *la ville de* proviennent d'un même type de phrase classificatrice :

Boston est une ville
Max est un salaud

Notons que la réduction de la phrase classificatrice contenant le nom *salaud* interdit la présence du déterminant du sujet :

Le facteur est un salaud ; il a couché avec ma femme
*= Ce salaud de (E + *le) facteur a couché avec ma femme*

[72] Un déterminant nominal très particulier qui pourrait s'appeler déterminant nominal locatif.
[73] La classe des noms appartenant à cette séquence est facilement listable : *connard, ignorant, énergumène*, etc.

Ce n'est pas le cas avec tous les noms propres de lieu :

Le Nord est un département ; il est connu pour son climat froid
*= le département de (*E + le) Nord est connu pour son pétrole*

On peut également se demander si les objets que nous étudions sont des noms composés. D'après G. Gross (1996), un nom composé a généralement la constitution syntaxique d'un groupe nominal libre : par exemple, *Det N Adj* = : *un pantalon bleu* (libre) + *un cordon bleu* (nom composé). Nos couples (*Nc, Npr*) ont clairement la même constitution qu'un groupe nominal libre :

- *Det N de Npr* =: *la vallée d'Aspe* [composé] + *la maison de Luc* [libre]
- *Det N de Det Npr* =: *le col du Somport* [composé] + *la pente du Vignemale* [libre]
- *Det N Npr* =: *la mer Méditerranée* [composé] + *le colonel Dupond* [libre]
- etc.

Par contre, les constituants syntaxiques d'un nom composé présentent un certain figement. Dans le meilleur des cas, le sens global du mot composé n'est pas compositionnel : il ne peut pas être calculé simplement à l'aide du sens des différents constituants. Cela apparaît très clairement avec le nom *cordon bleu* (excellent cuisinier). Mais ce n'est pas toujours le cas : *vin blanc* peut notamment être partiellement analysé à l'aide de la phrase classificatrice suivante :

Un vin blanc est un vin

Cette interprétation n'est pas valable pour *cordon bleu* (excellent cuisinier) ou *panier percé* (dépensier) :

** Un cordon bleu est un cordon*
** Un panier percé est un panier*

Traditionnellement, les noms composés du type *vin blanc* sont appelés noms composés endocentriques et ceux du type *cordon bleu* sont appelés noms composés exocentriques. Les couples (*Npr, Nc*) entrent clairement dans la première catégorie :

La mer (du Nord + Méditerranée) est une mer
Le mont (des Oliviers + Ventoux) est un mont

Dans certains cas, la phrase classificatrice n'est pas valide :

**la mer de Glace est une mer*

En effet, l'utilisation du classifieur *mer* est ici une métaphore lexicalisée. On peut également noter le cas d'*Ile de France* dont le classifieur est *région* et non *île* :

**l'Ile de France est une île*
l'Ile de France est une région

Le figement de certaines séquences est clair comme pour les expressions *mer du Nord* et *île du Diable*. Par ailleurs, la composition interne dépend de chaque couple et n'est pas toujours prédictible renforçant alors l'impression de figement de ces séquences :

*la mer (E + *d') Adriatique + la mer(*E + d')Aral*
=> l'(Aral + Adriatique) est une mer

*l'île (*E +de la) Réunion + l'île (E + *de) Maurice)*
=> (la Réunion + Maurice) est une île

D'autre part, certains couples (*Npr, Nc*) sont obligatoirement au pluriel[74] :

*(Les îles + *L'île) de les Canaries*
=> Les Canaries sont des îles.

Pour certains classifieurs *Nc*, le figement des couples (*Npr, Nc*) est moins net. Par exemple, les couples ayant pour classifieur *ville* ont un comportement prédictible si l'on connaît les propriétés syntaxiques propres à *Npr*. En effet, certains prennent des déterminants dont la première lettre est une majuscule[75] comme *Le Havre*, *La Havane* ou *Les Saisies*. Ces informations sont codées dans *Prolintex* sous la forme d'un trait syntaxique (+*DetLe* pour *Le Havre* ; +*DetLa* pour *La Havane* ; +*DetLes* pour *Les Saisies*). Les noms de villes ne prenant pas de déterminant comme *Toulouse* ont le trait syntaxique +*DetZ*.

la ville de (E + Det) Npr = : *la ville de (La Havane + Toulouse + Les Saisies)*

Dans ces exemples, *Npr* sélectionne un type de classifieur (*ville*) et non un autre (*mer* est interdit). Mais cette sélection lexicale est-elle suffisante pour considérer ces séquences comme des noms composés ? Le fait que l'on puisse insérer des modifieurs entre le nom *ville* et la préposition *de* fragilise cette thèse :

La ville musulmane de Téhéran
La ville francophone de Québec

De plus, ce type de comportement est habituellement décrit comme un figement mais avec la notion de nom approprié.

Pour finir cette discussion, nous évoquons un dernier point pouvant laisser penser que les objets que l'on traite sont des noms composés. En effet, il est bien connu que la présence d'un modifeur est interdite dans la séquence suivante :

*Max est en mer (E + *déchaînée)*

Par contre, la phrase suivante est parfaitement naturelle :

Max est en mer (Méditerranée + du Nord)

[74] Pour les îles, ces séquences au pluriel désignent des ensembles d'îles.
[75] On trouve des textes où ces déterminants n'ont pas de majuscule.

Cela montre que *Méditerranée* et *du Nord* ne sont pas des modifieurs classiques et que *mer Méditerranée* et *mer du Nord* pourraient former une unité. Par ailleurs, si l'on prend la *vallée d'Aspe*, on observe les situations suivantes :

** Luc est en vallée (E + fertile)*
Luc est en vallée de (Aspe + la Tarentaise)
** Luc est en vallée de (Aspe + la Tarentaise) fertile*

On a le même phénomène avec *école d'ingénieur* qui est clairement un nom composé.

**Max est en école (E + célèbre + mixte)*
Max est en école d'ingénieur
**Max est en école d'ingénieur célèbre*

Cette discussion est moins intéressante par son côté terminologique que par le fait qu'elle montre les particularités linguistiques des expressions que l'on traite. Il faut retenir de cette section l'existence claire de figements à différents niveaux : composition interne et détermination. Nous parlerons dorénavant de noms propres composés étendus que nous abrégerons en noms propres composés.

4.3.4 Composition syntaxique des formes longues et classification

Nous proposons maintenant d'examiner la structure interne des noms propres composés. De manière générale, la composition interne des noms composés est extrêmement variée :

- Adjectif Nom (AN) : *faux-cul, rouge gorge*, etc.
- Nom Adjectif (NA) : *cordon bleu, carte bleue*, etc.
- Nom de Nom (NDN) : *acte de foi, gardien de but*, etc.
- Nom à Nom (NAN): *panier à pain, moulin à vent*, etc.
- Nom Adjectif de Nom (NADN) : *offre publique d'achat*, etc.
- Etc.

Il en est de même pour les noms propres composés pour lesquels nous utilisons une notation similaire : *N* pour nom, *A* pour adjectif, *P* pour préposition, *D* pour déterminant, *Npr* pour nom propre (forme courte), *Npr-a* pour un adjectif morphologiquement dérivé du nom propre *Npr*. Jusqu'à présent, nous avons répertorié les formes longues suivantes :

- *LE Nc Npr* =: *la mer Méditerranée* (NNpr)
- *LE Nc Adj Npr* =: *l'océan glacial Arctique* (NANpr)
- *LE Nc Prep Npr* =: *l'état de Californie* (NPNpr)
- *LE Nc Prep Det Npr* =: *l'île à les Moines + le pic de le Midi* (NPDNpr)
- *LE Nc Npr-a*[76] =: *la République française* (NNpr-a)
- *Le Nc Adj de Npr* =: *la République arabe d'Égypte* (NAPNpr)
- *LE Nc Adj de Det Npr* =: la *République démocratique de le Congo* (NAPDNpr)

Il existe des noms officiels de pays ayant des structures syntaxiques internes encore plus complexes :

- *LE Nc Adj et Adj de Npr* = : *la République populaire et démocratique de Corée*

[76] *Npr-a* est l'adjectif morphologiquement lié à *Npr* (ex : pour *Npr* = : *France*, *Npr-a* = : *française*).

- *LE Nc Adj Npr-a Adj et Adj= : la Jamahiriya arabe libyenne populaire et socialiste*

La très grande majorité des noms propres de lieu rentrent dans au moins une des trois structures suivantes : NNpr, NPNpr ou NPDNpr. Les structures contenant un adjectif concernent essentiellement les noms de pays dont les formes officielles courtes et longues sont répertoriées à partir de la liste diffusée par la Délégation Générale à la Langue Française[77]. Ils ont également été étudiés par O. Piton et al. (1997) qui, à partir de cette liste, ont systématiquement regardé leur comportement dans un texte journalistique. Nous examinons, pour l'instant, les formes *LE Nc ((de) Det) Npr* où les séquences entre parenthèses sont optionnelles. Ces séquences sont très largement majoritaires dans l'ensemble des noms propres de lieu. Comme nous l'avons mentionné précédemment, la plupart des noms propres rentrent dans au moins l'une de ces structures :

mont Ventoux, mer Méditerranée, île Maurice (NPDNpr)
île de Malte, col de Splandelle, mer de Barentz (NPNpr)
département de la Gironde, col de le Tourmalet (NPDNpr)

Les déterminants *Det* de la forme longue de type NPDNpr sont limités aux déterminants définis suivants : *le*, *la* et *les*. Nous donnons ci-dessous quelques exemples :

l'île de (la Barbade + le Diable)
l'état de (le Texas + la Californie)
le col de (le Tourmalet + la Colombière)
le mont de les Oliviers

Les autres déterminants sont exclus. Par exemple, les séquences suivantes sont clairement interdites :

** L'île de (cette + sa + une) Barbade*
** l'état de (ce + son + un) Texas*
** le mont de (ces + ses + des) Oliviers*

Certains couples rentrent dans plusieurs structures équivalentes. Dans plusieurs noms propres dont la forme longue comprend la préposition *de*, le déterminant *Det* est optionnel :

La principauté de (E + le) Liechtenstein
L'état de (le + E) Vermont[78]
L'état de (le + E) Washington[79]

Parfois, c'est la préposition *de* qui est facultative :

Les îles (E + de les) Canaries
Le désert (de + ?E)Mojave[80]

Dans certains cas comme pour le couple (*Ile-de-France, région*), les trois structures internes sont possibles :

[77] http://www.culture.fr/culture/dglf/ressources/pays/pays.htm.
[78] La forme *état de Vermont* trouvée dans le Monde 1994 (1 occurrence).
[79] La forme surprenante *état du Washington* aussi trouvée dans le Monde 1994 (2 occurrences).
[80] La forme *désert Mojave* a été trouvée dans le Monde 1994 (1 occurrence).

La région (de la + de + E) Ile-de-France

Certains noms propres n'ont pas de forme longue associée à *Npr*, c'est le cas de *Manche* et *Canada* :

La Manche est une mer
** la mer (E+ de + de la) Manche*

Le Canada est (un pays + un état)
** l'état du Canada*

Pour certains, on peut facilement associer un classifieur : la *Manche* est clairement une mer ; le *Canada* est à la fois un pays, un état, etc. et il est très difficile de faire un choix clair.

Ces premières constatations générales convergent déjà vers la difficulté (voire l'impossibilité) de prévoir la composition syntaxique de la forme longue d'un nom propre, étant donné son couple (*Npr*, *Nc*). Une étude systématique est donc nécessaire. Etant donné la quantité astronomique de noms propres de lieu à répertorier, il est nécessaire de les classer. Une première solution consiste à regrouper les noms propres selon leur structure interne et plus exactement leur structure interne la plus longue. Par exemple, les couples (*Réunion*, *île*), (*Ile-de-France*, *région*), (*Vermont*, *état*) et (*Pirée*, *ville*) seraient insérés dans la classe NPDNpr car leurs formes les plus longues sont de ce type :

L'île de la Réunion
La région (de l' + de + E) Ile-de-France
L'état (de le + de) Vermont
La ville de le Pirée

Le couple (*Iran*, *république*) serait intégré à la classe NAPNpr car sa forme la plus longue contient l'adjectif approprié *islamique* : *la république islamique d'Iran*. Le gros avantage de cette classification est qu'elle est très facile à mettre en place : construction d'une table par structure syntaxique. Ainsi, la classification de formes rares comme *Jamahiriya arabe libyenne populaire et socialiste* ne pose pas de problème. Etant donné un couple et sa forme la plus longue, l'ajout est immédiat. Cependant, cette méthode présente de multiples inconvénients. Tout d'abord, l'intuition linguistique sur les noms propres de lieu n'est pas toujours très fiable. L'acceptabilité de certaines structures peu connues est surtout basée sur les tendances générales, sur des attestations trouvées dans les corpus de travail[81] ou sur des intuitions phonologiques. Ainsi, la classification pour beaucoup de noms propres serait basée sur une sorte d' « approximation »[82]. D'autre part, certaines structures dans lesquelles rentrent certains noms propres sont plutôt marginales dans l'ensemble des noms propres, mais spécifiques de certains classifieurs. Tout d'abord, les noms des océans ont tous la structure NNpr (*océan Atlantique*, *océan Pacifique*, *océan Indien*), sauf un qui comprend un adjectif approprié facultatif entre le classifieur *océan* et *Npr*[83] : *océan (glacial + E) Arctique*. Ensuite, certains classifieurs comme *département*, *république* ou *état* sont très enclins à accepter des

[81] Une occurrence d'une expression dans un texte peut être une erreur de l'auteur ; l'absence d'une expression dans un texte ne prouve pas son inacceptabilité.
[82] On peut remarquer que ces problèmes épistémologiques sont généraux et non spécifiques à cette classification. Cependant, ils sont amplifiés par la spécificité du domaine des noms propres.
[83] Pour les océans, le choix de catégoriser Atlantique comme *Adj* ou *Npr* est arbitraire.

structures adjectivales, ce qui n'est pas le cas de la très grande majorité des classifieurs :

Le département (landais + des Landes)
L'état (de Californie + californien)
La région (de l'Ile-de-France + francilienne)
La république (française + ?de France[84]*)*

Puis, un classifieur tel que *île* possède une particularité : un ensemble d'îles est souvent considéré comme un archipel et l'on observe des formes telles que

L'archipel de (*les îles de + E*) *les Açores*

Les observations générales précédentes mettent en évidence la grande hétérogénéité du comportement des noms propres, même ceux possédant le même classifieur. Cette constatation est vraie pour un certain nombre de classifieurs comme *mer*, *île* ou *république* :

*La mer (de le + *de + *E) Nord*
*La mer (*de le + de + *E) Barentz*
*La mer (*de la + *de + E) Méditerranée*

*L'île (*de + E) Maurice*
L'île (de + ?E) Malte
*L'île (*E + de la + *de) Martinique*
*Les îles (E + de les + *de) Canaries*

*Le mont (*de + *de le + E)Ventoux*
*Le mont (*de + de les + *E) Oliviers*

*La république (française + ?*de France)*
La république islamique de Iran
La république démocratique et populaire de Corée

Cependant, excepté pour quelques classifieurs comme *état* (au sens de pays) ou *république*, l'ensemble des structures possibles des formes longues est limité à NNpr, NPNpr et NPDNpr (voire NNpr-a), ce qui ne pose pas de problème de représentation dans une table syntaxique. Par ailleurs, pour quelques classifieurs tels que *ville*, *département*, *état* (partie administrative d'un pays), le nombre de structures est clairement limité. Par exemple, le classifieur *ville* n'accepte que les formes longues comprenant la préposition *de* (NPNpr et NPDNpr) :

?**la ville (Paris + Le Havre) est un haut-lieu touristique*
La ville de (Paris + Le Havre) est un haut-lieu touristique
? la ville (parisienne + havraise) est un haut-lieu touristique*

Pour le type NNpr-a, il semble que si l'on remplace le nom *ville* par *cité*, alors la dernière phrase devient plus naturelle :

La cité parisienne est un haut-lieu touristique

[84] Le nom propre *République de France*, bien qu'il nous paraisse interdit, a été trouvé sur Internet via le moteur de recherche *Google*.

Nous donnons ci-dessous un tableau montrant les restrictions sur les structures syntaxiques dans lesquelles peuvent apparaître les classifieurs *ville*, *état* et *département* :

Nc	NNpr	NPNpr	NPDNpr	NNpr-a
ville	-	+	+	-
état	-	+	+	+
département	-	+	+	+

Table 11 : comportement de classifieurs

Nous proposons donc de classer les noms propres selon leur nom classifieur. Cette méthode présente de nombreux avantages. La classification d'un nom propre est immédiate et n'est pas sujette à un risque d'erreur. La distribution des noms propres est mieux répartie, même s'il existe des classes comme les noms d'océans qui sont très petites par rapport à d'autres telles que celle des villes[85]. Le nombre de colonnes des tables sera exactement adapté aux besoins, évitant ainsi d'avoir des parties creuses dans les tables, comme cela aurait pu être le cas dans des tables classées selon la structure syntaxique interne des noms propres donc sémantiquement moins spécifiques. Jusqu'à présent, nous avons travaillé sur une soixantaine de classifieurs. On pourrait reprocher à cette méthode de générer un très grand nombre de tables. Mais, vu le nombre d'entrées potentielles, ce n'est pas un problème, plutôt un avantage. Nous avons dénombré deux gros défauts à cette approche. Tout d'abord, il existe des noms propres qui n'ont pas de formes longues et dont le classifieur n'est pas clair (*Canada*, *Roumanie*, etc.). La solution est de construire une table résiduelle contenant ce genre d'entrées. Ensuite, quelques noms propres comme *mer de Glace* ne rentre pas dans la phrase classificatrice *Detc Nprc être (UN + des) Nc* comme le font la quasi-totalité des noms propres de lieu :

** La mer de Glace est une mer*
La mer de Corée est une mer

Dans cet exemple, la *mer de Glace* n'est pas une mer, mais plutôt un glacier (dans les Alpes). Cet emploi est uniquement métaphorique. Si l'on place cette entrée dans la même classe, il y a une hétérogénéité sémantique. Mais, cela n'est pas très dérangeant car il suffit de rajouter une colonne dans la table des noms de mers désignant cette déviation sémantique.

De manière générale, le processus de classification et d'étude systématique d'objets linguistiques est freiné par les différents emplois que peuvent avoir certains mots de la langue. Notre étude ne fait pas exception à la règle. D'abord, il existe différents emplois pour certains noms propres *Npr*[86]. Par exemple, *Nord* est soit une mer soit un département français. Dans ce cas-là, la distinction est évidente car ils n'ont pas le même classifieur (*mer* et *département*) et sont classés dans deux tables différentes. Il existe un cas très difficile à traiter : les lieux qui ont le même nom et le même classifieur comme *Paris* (*ville*) qui est à la fois une ville de France et une ville des Etats-Unis (et peut-être même ailleurs). Il n'est malheureusement pas possible de distinguer ces deux entrées dans nos tables, sauf si l'on ajoute des propriétés extra-linguistiques reliées à la position géographique des lieux que l'on traite (et l'on sort de notre cadre de travail). Il existe également différents emplois pour les noms classifieurs. Certains d'entre eux sont ambigus. Il existe deux types d'ambiguïtés : le classifieur ambigu peut avoir

- deux emplois du type *Nc* faisant partie d'un nom propre *Nprc* (par exemple, *état* [pays

[85] Il existe des centaines de milliers de ville dans le monde et seulement quatre océans.
[86] Cf. également O. Piton et D. Maurel (2001).

ou région administrative] ou *côte* [bord de mer ou pente])
- un emploi de type *Nc* et un emploi n'entrant pas dans notre étude (ex : *région* [zone administrative ou zone approximative]).

Les premiers cas sont distingués par des critères syntaxiques (ex : *côte* cf. section sur la distribution prépositionnelle) ou simplement par notre connaissance du monde (ex : *état*).
Examinons maintenant le deuxième cas et prenons l'exemple de *région* :

la région du Havre
la région du Nord-Pas-de-Calais

Le premier exemple désigne la région autour du Havre et ne rentre pas dans notre étude; la seconde désigne la région administrative du Nord-Pas-de-Calais et rentre dans notre étude. Un premier moyen de les distinguer syntaxiquement est d'utiliser la phrase classificatrice :

** Le Havre est une région*
Le Nord-Pas-de-Calais est une région

Le nom *région* du premier cas pourrait être rattaché aux noms de localisation spatiale au sens d'A. Borillo (1989) (cf. section sur les prépositions locatives composées). En effet, l'expression *la région de* peut être remplacée par *les alentours de* où *alentour* est un nom de localisation externe :

Max a bien connu la région du Havre
?= Max a bien connu les alentours du Havre

Ainsi, si l'on pousse l'analyse plus loin, on peut considérer l'expression *dans la région de* comme une préposition locative dans la phrase suivante car elle peut également être remplacée par la préposition locative composée *près de* :

Marie habite (*dans la région de + dans les alentours de + près de) (la ville du + Le) Havre*

L'analyse précédente n'est clairement pas valable pour le deuxième cas :

Marie a bien connu la région du Limousin
≠ Marie a bien connu les alentours du Limousin

4.3.5 Réduction des formes longues et figement

Les formes longues des noms propres (*Nprc*) peuvent être réduites en formes courtes (*Npr*) par le processus suivant. Tout d'abord, la séquence *LE Adj* Nc Adj* (de)*[87] est effacée dans *Nprc*. Puis, s'il ne se trouve pas explicitement dans la forme longue, le déterminant *Det* est ajouté à gauche de cette nouvelle séquence. Ce déterminant peut être vide (*Det* = : *E*).

le département de les Pyrénées-Atlantiques
= les Pyrénées-Atlantiques (déterminant *les* explicite)

[87] La séquence *(de)* signifie que *de* est optionnel. Le symbole * est le symbole de Kleene : *a** désigne l'expression régulière *(E + a + aa +aaa + …)*.

la ville de Paris
= Paris

la république de Hongrie
*= *Hongrie*
= la Hongrie

le Mont Ventoux
*= *Ventoux*
= le Ventoux

? le fleuve Seine
*= *Seine*
= la Seine

Les jugements d'acceptabilité dans les exemples ci-dessus se réfèrent à des phrases du type

Ceci est Nprc
=: Ceci est (la ville de Paris + Paris)

Les déterminants définis ne sont pas les seuls à être autorisés avec les formes courtes. On trouve également des déterminants possessifs avec un sens affectif très expressif :

sa république du Mali => son Mali
sa mer Méditerranée => sa Méditerranée
son mont Ventoux => son Ventoux
son fleuve Seine => sa Seine

Les noms propres *Npr* ayant le déterminant *Det* vide acceptent sans difficulté particulière les déterminants possessifs (K. Jonasson, 1995 ; D. Maurel et O. Piton, 1998) :

sa ville de Paris => son Paris
son île de Tahiti => son Tahiti

De même, ces séquences acceptent des modifieurs : *sa belle ville de Paris* devient *son beau Paris*. Ces variantes sont très connues (voir K. Jonasson, 1995).

La réduction des formes longues en formes courtes n'est pas régulière. Il arrive que la forme courte de certains couples soit différente de *Npr*. Par exemple, la forme courte de *république démocratique populaire de Corée* est *Corée du Nord* et non *Corée*. Ce phénomène est limité car il ne touche quasiment que les noms de pays. Ensuite et surtout, certaines formes longues très figées ne peuvent être réduites. Nous donnons ci-dessous quelques d'exemples :

*(la mer de + *E) le Nord*
*(la mer de + *la) Corée*
*(la mer + *la) Noire*
(*le pic de + *E) le Midi*
*(l'île de + *E) la Tortue*

Notons que, dans certains textes, on trouve le classifieur de certaines formes longues figées,

avec la première lettre en majuscule : *la Vallée de la Mort*.

Le figement peut servir notamment à distinguer des emplois ambigus de *Npr*. Par exemple, *Nord* est soit un département soit une mer. Ces deux entrées se distinguent non seulement par leur classifieur (donc ils sont dans deux tables différentes), mais aussi par leur comportement syntaxique. En effet, *la mer du Nord* n'a pas de forme courte alors que *le département du Nord* est clairement réductible à la forme courte *le Nord*.

Nous nous attachons maintenant à montrer les différents degrés de figement dans *Nprc*. D'abord, le figement peut être distributionnel comme pour le nom *ville*. Tous les noms de villes entrent dans la construction nominale suivante :

La ville de Npr =: la ville de (Paris + Le Havre)

Cette analyse est possible, même si certains noms de villes possèdent des déterminants (ex : *Le Havre*, *La Havane*, *Les Saisies*) ; ces derniers ayant souvent leur première lettre en majuscule, l'ensemble peut être vu comme un *Npr*. Ainsi, la reconnaissance de la forme longue composée requiert seulement une bonne grammaire de *Npr* (cf. applications). On retrouve aussi ces déterminants écrits sans majuscule. Lorsque le déterminant *Le* est précédé de la préposition *à* ou *de*, la séquence est contractée en *au* ou en *du* (équivalent avec *à le* ou *de le*) : *à Le Havre = au Havre = à le Havre*. Par conséquent, il est préférable de considérer le déterminant en dehors de *Npr*. Ainsi, nous avons la structure

La ville de (Det+E) Npr

Linguistiquement, le choix de considérer *Det* comme inclus ou non dans *Npr* est arbitraire ; ce choix peut être guidé par des considérations sur les applications.

Il est alors absolument nécessaire de regarder chaque nom de ville et lui assigner, quand cela est nécessaire, un déterminant. C'est ce qui a été réalisé dans *Prolintex*.

Pour certains classifieurs comme *état* (état américain), les couples (*Nc*, *Npr*) semblent obéir à quelques règles approximatives. Etant donné le déterminant *Det* utilisé, il est possible de produire la forme longue associée :

- si le déterminant *Det* est *le*, le couple (*Npr*, *Nc*) accepte la construction nominale

  l'état de Det Npr =: *l'état de le Texas*

- Si le déterminant *Det* est *la* ou s'il est vide, le couple forme un nom composé de la forme :

  l'état de Npr =: l'état de (New York + Californie)

- si le déterminant Det est l', les deux sont possibles :

  l'état de (E + l') Npr =: l'état de (E + l') Oregon

Comme précédemment, la génération de telles formes composées nécessite l'association systématique d'un déterminant défini à chaque nom d'état :

Californie => la
Orégon => l'
Texas => le
New-York => E

Ces règles sont approximatives : il existe des exceptions. Par exemple, la forme *état de le Vermont*, bien qu'il sélectionne le déterminant *le*, possède une autre variante : *l'état de Vermont*.
Par ailleurs, un classifieur comme *département* restreint la structure des noms composés à *LE Nc de Det Npr*. La seule variation vient du déterminant *Det* qui dépend de *Npr* :

le département de (le Nord + la Picardie + les Landes)

Il existe cependant une exception avec le *Territoire de Belfort* dont la forme composée paraît douteuse :

? Lundi dernier, j'ai visité le département du Territoire de Belfort*

4.3.6 Les noms propres composés dans les groupes nominaux

Nous examinons comment se comportent les noms propres composés locatifs dans les groupes nominaux. Nous regardons d'abord la détermination en distinguant les emplois singuliers des emplois pluriels, puis nous étudions la distribution des modifieurs.

4.3.6.1 Détermination : les emplois singuliers

Lorsqu'ils possèdent une forme courte, les couples ayant un emploi singulier entrent dans la construction

*(E + Det) Npr être (UN + *des) Nc*

*Paris est une ville / * Paris (sont + E) des villes*
*La Réunion est une île / * la Réunion (sont + E) des îles*

Dans le groupe nominal issu de la phrase, on notera Det_c le déterminant de *Nc* : Det_c *Nc Npr*, Det_c *Nc de Npr*, Det_c *Nc de Det Npr*, etc.

La distribution de Det_c dans les constructions nominales dérivées est complexe. Le déterminant le plus naturel est le déterminant défini LE (*le*, *la*) :

la ville de Pau m'a toujours fasciné
la région (E + de le) Nord-Pas-de-Calais attire beaucoup de touristes belges

Le déterminant indéfini UN et le déterminant démonstratif *ce* sont quant à eux interdits :

** Marie apprécie de revoir (une + cette) ville de Paris*
** Marie adore (un + cet) état de la Californie*

Les déterminants démonstratifs *un* et *ce* avec un modifieur sont acceptables :

*Max a pu observer une ville de Paris (*E + dévastée par les bombes)*

*J'apprécie de revoir cette (bonne vieille + ?*E) ville de Paris (E + que j'aime tant)*

Le démonstratif *ce* sans modifieur semble acceptable dans quelques rares cas mais produit l'impression d'une ellipse :

Marie me casse les pieds avec cette (?E + satanée) avenue des Champs-Elysées

Notons que le déterminant possessif est parfaitement acceptable mais avec une interprétation particulière :

Max aime parler de son île de la Guadeloupe
≠ Max aime parler de sa maison

Cette distribution est similaire pour les formes figées :

*Sophie aime parler de (la + sa + *cette + *une) mer (du Nord + Morte)*
Sophie a vu une mer Morte déchaînée par les vents tourbillonnants
Sophie déplore une triste mer du Nord

4.3.6.2 Détermination : les emplois pluriels

Il existe deux cas d'emplois pluriels. Tout d'abord, nous examinons celui où le couple (*Nc, Npr*) rentre dans la construction

(E + Det) Npr être (UN + des) Nc*
Les Shetland sont (une + des) îles*

Nous retrouvons alors la même distribution de Det_c (transposées au pluriel) dans les groupes nominaux :

*Max décrit (les + ?ses + ?*ces + *des) îles Canaries*
*Luc me casse les pieds avec (les + ses + ? ces + *des) îles Shetland*

Dans ces exemples, nous avons affaire à des regroupements d'îles qui s'identifient par leur emploi pluriel. Les noms composés tels que *les îles Marshall* fonctionnent de la même manière.

Le deuxième cas correspond à la coordination de couples (*Nc, Npr*). Le groupe nominal pluriel

les villes de Paris, (de + E) Lyon et (de + E) Marseille

est en fait la factorisation de trois groupes nominaux singuliers :

la ville de Paris ; la ville de Lyon ; la ville de Marseille

Par ailleurs, nous constatons l'apparition dans les textes de groupes nominaux pluriels comme

les villes Paris, Lyon et Marseille ont conclu un accord commercial

alors que l'emploi singulier est exclu :

? La ville Paris est candidate à l'organisation des JO*

Ces coordinations sont très naturelles dans les constructions nominales contenant la préposition *de* :

Les royaumes de Belgique et du Danemark ne sont pas très éloignés l'un de l'autre.
Les îles de Sardaigne et de Sicile attirent beaucoup de touristes.

Les coordinations de formes nominales sans préposition *de* sont aussi possible :

Max aime survoler les mers Méditerranée et Adriatique

Par contre, la coordination entre constructions de différentes structures est plus difficile :

? Max déteste les îles Maurice et de la Réunion*

La factorisation du classifieur dans une coordination de noms propres composés figés est acceptable, même si elle n'est pas très naturelle :

?Luc a déjà affronté les mers de Corée et du Nord
?Luc a déjà affronté les mers Morte et Noire
?Marie a traversé les vallées de la Mort et des Rois

Ces constructions sont plus naturelles avec des noms propres moins figés :

Marie a traversé les vallées de Chevreuse et de la Maurienne.

4.3.6.3 Les modifieurs autour de *Nc*

Nous examinons la distribution des modifieurs dans les groupes nominaux étudiés précédemment. Les groupes nominaux peuvent avoir les formes suivantes où *M1*, *M2* et *M3* sont trois positions de modifieurs :

Det_c M1 Nc M2 de (E + Det) Npr M3
Det_c M1 Nc Npr M3

Le département très montagneux des Pyrénées-Atlantiques
La grande gare Montparnasse

La position M2 dans le groupe nominal sans préposition *de* est interdite :

La mer très polluée qu'est la Méditerranée
la mer très polluée Méditerranée
la très polluée mer Méditerranée

La position M2 est stylistiquement réservée à des modifieurs très courts :

La ville où Marie a vécu toute son enfance de Venise
La ville inoubliable de Venise

On retrouve les modifieurs longs à la position M3 :

La ville de Venise, où Marie a vécu toute son enfance

Le cas des noms propres composés figés est quasiment similaire. La seule différence réside dans le fait que l'insertion d'un modifieur en position M2 est la plupart du temps totalement interdite :

*La mer (E + *agitée) du Nord attire beaucoup de touristes*
*Max escalade le pic (E + *dangereux) du Midi*

4.3.7 Codage des contraintes internes

4.3.7.1 Représentation sous la forme de tables

Dans les sections précédentes, nous avons montré un ensemble de contraintes au sein des noms propres composés. Nous les codons maintenant dans des tables syntaxiques où chaque ligne correspond à une entrée lexicale (i.e. un nom propre composé *Nprc*). Nous construisons une table pour chaque classe de *Nprc*, c'est-à-dire pour chaque classifieur. Soit un classifieur *X*, alors sa table syntaxique sera nommée **NNpr-X**. Par exemple, la table associée au classifieur *mer* est la table **NNpr-mer**. Lorsque le classifieur est ambigu comme *état*, nous précisons l'indice associé à cet emploi. Par exemple, nous aurons la table **NNpr-état-69** pour *état* au sens de « région administrative » (69 est l'indice associé à cet emploi). La table **Npr** est la table décrivant l'ensemble des noms propres de lieu n'ayant pas de forme longue et dont le classifieur ne peut être exactement précisé. Jusqu'à présent, nous avons entamé le codage d'une soixantaine de tables. Certaines comme les départements (français), les états américains, les provinces canadiennes ou les régions (françaises) sont complètes. Dans la suite, nous expliquons le contenu de quelques tables codées.
Avant tout, nous expliquons les intitulés des colonnes utilisés. Les noms ont été choisis de telle manière qu'ils soient explicites pour le lecteur. Les principaux intitulés sont synthétisés et expliqués dans le tableau ci-dessous :

Intitulé	**Propriété ou information lexicale**
Nc	Classifieur
Npr	Nom propre associé au classifieur
Detc pluriel	Le nom propre composé est au pluriel (ex : *les îles Marshall*)
Prep	Préposition
Det	Déterminant défini associé à *Npr*
Npr-a	Adjectif morphologiquement lié à *Npr*
Adj1, *Adj2*, *Adj3*	Adjectifs de (*Npr*, *Nc*)

Certains intitulés sont des structures dans lesquels les noms propres peuvent rentrer : *LE Nc de Npr*, *LE Nc Npr-a*, etc. Par ailleurs, à chaque entrée lexicale, nous associons un numéro dans la colonne intitulée *Index Npr*. De même, chaque classifieur possède un numéro unique (colonne *Index Nc*). Dans la suite, nous décrivons quelques tables qui sont ordonnées selon le degré de complexité. Nous donnons des extraits de chacune d'elles. Dans les extraits de tables que nous proposons, les classifieurs peuvent apparaître redondants mais leur présence rend la lecture plus facile.

4.3.7.2 Table Npr

La table *Npr* est la plus simple et elle contient les noms propres qui ne possèdent pas de forme longue et de classifieur clair tels que le *Canada*. Cette table comprend surtout des noms de pays.

A	B	C	D
Indcx Npr	Dct	Npr	Variantes Npr
8	<E>	Antigua-et-Barbuda	-
6	la+l'	Australie	-
9	la	Barbade	-
10	le	Belize	-
11	le	Burkina Faso	Burkina
12	le	Canada	-
28	la	Grande-Bretagne	-
14	la	Grenade	-
15	la+l'	Irlande	-
29	la+l'	Irlande de le Nord	-
16	la	Jamaïque	-
17	le	Japon	-
18	<E>	Kiribati	-
34	le	Labrador	-
19	la	Malaisie	-
20	la	Mongolie	-
31	le	Negara Brunei Darussalam	Brunéi Darussalam+Brunei
21	la	Nouvelle-Zélande	-
22	la	Papouasie-Nouvelle-Guinée	Papouasie
35	le	Pays basque	-
30	la	Roumanie	-
23	<E>	Sainte-Lucie	-
24	<E>	Saint-Vincent-et-les Grenadines	-
33	<E>	Terre-Neuve	-
25	le	Turkménistan	-
26	<E>	Tuvalu	-
27	la+l'	Ukraine	-

Table 12 : échantillon de la table Npr

4.3.7.3 Table NNPr-département

A	B	C	D	E	F	G	H	I
Index Npr	Index Nc	Nc	Prep	Det	Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a
1	8	département	de	le+l'	Ain	-	-	-
2	8	département	de	le+l'	Aisne	-	-	-
3	8	département	de	le+l'	Allier	-	-	-
4	8	département	de	les	Alpes-de-Haute-Provence	-	-	-
5	8	département	de	les	Hautes-Alpes	-	-	-
6	8	département	de	les	Alpes-Maritimes	-	-	-
7	8	département	de	le+l'	Ardèche	+	ardéchois	+
8	8	département	de	les	Ardennes	-	ardennais	+
9	8	département	de	le+l'	Ariège	-	ariégeois	+
10	8	département	de	le+l'	Aube	-	aubois	+
11	8	département	de	la+l'	Aude	-	audois	+
12	8	département	de	le+l'	Aveyron	-	aveyronnais	+
13	8	département	de	les	Bouches-de le-Rhône	-	-	-
14	8	département	de	le	Calvados	-	-	-
15	8	département	de	le	Cantal	-	cantalien+cantalou	+
16	8	département	de	la	Charente	+	charentais	+
17	8	département	de	la	Charente-Maritime	+	-	-
18	8	département	de	le	Cher	-	-	-
19	8	département	de	la	Corrèze	+	corrézien	+
20	8	département	de	la	Corse-de le-Sud	+	-	-
21	8	département	de	la	Haute-Corse	+	-	-
22	8	département	de	la	Côte-de or	-	-	-

Table 13 : échantillon de la table NNpr-département

4.3.7.4 Table NNpr-mer

A	B	C	D	E	F	G	H	I	J	K	L	M
Index Npr	Index Nc	Nc	Modif M2 optionnel	Prep	Det (forme longue)	Npr	Variante Npr	LE Nc de Npr	LE Nc Npr	(Detc Nprc+Det Npr) être un Nc	Det (forme courte)	Det Npr
1	19	mer	-	-	-	Adriatique	-	-	+	+	l'+la	+
21	19	mer	+	de	les	Antilles	-	-	-	+	-	-
2	19	mer	-	-	-	Baltique	-	-	+	+	la	+
11	19	mer	+	de	-	Barentz	-	+	-	+	-	-
3	19	mer	-	-	-	Blanche	-	-	+	+	-	-
22	19	mer	+	de	les	Caraïbes	-	-	-	+	-	-
12	19	mer	+	de	-	Chine	-	+	-	+	-	-
13	19	mer	+	de	-	Chine Méridionale	Chine de le Sud	+	-	+	-	-
14	19	mer	+	de	-	Corée	-	+	-	+	-	-
15	19	mer	+	de	-	Crète	-	+	-	+	-	-
4	19	mer	-	-	-	Egée	-	-	+	+	-	-
23	19	mer	-	de	le+l'	Est	-	-	-	+	-	-
16	19	mer	-	de	-	Glace	-	+	-	-	-	-
5	19	mer	-	-	-	Intérieure	-	-	+	+	-	-
17	19	mer	+	de	-	Irlande	-	+	-	+	-	-
24	19	mer	+	de	le	Japon	-	-	-	+	-	-
6	19	mer	-	-	-	Jaune	-	-	+	+	-	-
18	19	mer	+	de	-	Java	-	+	-	+	-	-
19	19	mer	+	de	-	Kara	-	+	-	+	-	-
27	19	mer	-	-	-	Manche	-	-	-	+	la	+
7	19	mer	-	-	-	Méditerranée	-	-	+	+	la	+
8	19	mer	-	-	-	Morte	-	-	+	+	-	-
9	19	mer	-	-	-	Noire	-	-	+	+	-	-
25	19	mer	-	de	le	Nord	-	-	-	+	-	-

Table 14 : échantillon de la table NNpr-mer

Quelques remarques de lecture de la table :
Lorsque la case correspondant à Prep est -, cela signifie que la structure LE Nc Prep Det Npr est interdite. Il en est de même avec les éléments de la colonnes Det (forme longue).

4.3.7.5 Table *NNpr-île*

La table **NNpr-île** est un peu plus compliquée. La colonne *Detc pluriel* indique si le nom propre composé est au pluriel : *les îles Bahamas* / * *l'île Bahamas*. La colonne intitulée *LE archipel de Det Npr* indique si le nom propre composé a une variante de cette forme :

Les îles Bermudes
= l'archipel des Bermudes

Cette propriété est valable dans les cas où le classifieur est *île* et où le nom propre est obligatoirement au pluriel, ce qui n'est pas une surprise car un archipel est un ensemble de plusieurs îles.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Index Npr	Detc pluriel	Index Nc	Nc	Modif M2	Prep	Det	Npr	LE Nc de Npr	LE Nc Npr	LE archipel de Det Npr	Npr-a	LE Nc Npr-a	Det (forme courte)	Det Npr
1	+	17	île	-	-	-	Aléoutiennes	-	+	+	-	-	les	+
56	-	17	île	-	de	la	Ascension	-	-	+	-	-	la	-
2	+	17	île	-	-	-	Bahamas	-	+	+	bahamien	+	les	+
3	+	17	île	+	de	les	Baléares	-	+	+	-	-	les	+
57	-	17	île	+	de	la	Barbade	-	-	-	barbadien	+	la	+
34	-	17	île	-	de	-	Beauté	+	-	-	-	-	-	-
4	+	17	île	-	-	-	Bermudes	-	+	+	-	-	les	+
35	-	17	île	+	de	-	Bornéo	+	-	-	-	-	<E>	+
5	+	17	île	+	de	les	Canaries	-	+	+	-	-	les	+
58	+	17	île	+	de	le	Cap-Vert	-	-	-	cap-verdien	+	le	+
6	+	17	île	-	-	-	Caraïbes	-	+	+	-	-	les	+
7	+	17	île	-	-	-	Célèbes	-	+	+	-	-	les	+
36	-	17	île	+	de	-	Ceylan	+	-	-	-	-	<E>	+
37	-	17	île	+	de	-	Chypre	+	-	-	chypriote	+	<E>	+
38	-	17	île	+	de	-	Corse	+	-	-	corse	+	la	+
39	-	17	île	+	de	-	Crète	+	-	-	crétois	+	la	+
40	-	17	île	+	de	-	Cuba	+	-	-	cubain	+	<E>	+
59	-	17	île	+	de	le	Diable	-	-	-	-	-	le	-
41	-	17	île	+	de	-	Elbe	+	-	-	-	-	<E>	+
8	+	17	île	-	-	-	Féroé	-	+	-	-	-	les	+

Table 15 : échantillon de la table NNpr-île

4.3.7.6 Table *NNpr-république*

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
Index Npr	Index Nc	Nc	Adj1	Prep	Det	Npr	LE Nc Adj1 de Det Npr	LE Nc Adj1 de Npr	LE Nc de Det Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det	Variante Det	Det Npr	Variante Npr
106	30	république	-	-	-	-	-	-	-	-	dominicaine	+	-	-	-	-
124	30	république	socialiste	de	le	:Vietnam	+	-	+	-	vietnamien	-	le	-	+	-
1	30	république	-	de	-	Afrique de le Sud	-	-	-	+	sud-africain	+	la+l'	-	+	-
2	30	république	-	de	-	Albanie	-	-	-	+	albanais	+	l'+la	-	+	-
107	30	république	fédérale	de	-	Allemagne	-	+	+	-	allemand	-	l'+la	-	+	-
4	30	république	-	de	-	Angola	-	-	-	+	angolais	+	le+l'	-	+	-
95	30	république	-	-	-	Argentine	-	-	-	-	argentin	+	la+l'	-	+	-
5	30	république	-	de	-	Arménie	-	-	-	+	arménien	+	la+l'	-	+	-
6	30	république	-	de	-	Autriche	-	-	-	+	autrichien	+	la+l'	-	+	-
3	30	république	-	de	-	Azerbaïdjan	-	-	-	+	-	-	le+l'	-	+	-
117	30	république	populaire	de	le	Bangladesh	+	-	+	-	-	-	le	-	+	-
53	30	république	-	de	le	Bélarus+Belarus	-	-	+	-	-	-	le	-	+	-
7	30	république	-	de	-	Belau	-	-	-	+	-	-	<E>	-	+	-
51	30	république	-	de	le	Bénin	-	-	+	-	-	-	le	-	+	-
52	30	république	-	de	le	Bhoutan	-	-	+	-	-	-	le	-	+	-
8	30	république	-	de	-	Biélorussie	-	-	-	+	biélorusse	+	la	-	+	-
9	30	république	-	de	-	Bolivie	-	-	-	+	bolivien	+	la	-	+	-
10	30	république	-	de	-	Bosnie-Herzégovine	-	-	-	+	bosniaque	-	la	-	+	-
54	30	république	-	de	le	Botswana	-	-	+	-	-	-	le	-	+	-
118	30	république	fédérative	de	le	Brésil	+	-	+	-	brésilien	-	le	-	+	-
11	30	république	-	de	-	Bulgarie	-	-	-	+	bulgare	+	la	-	+	-
55	30	république	-	de	le	Burundi	-	-	+	-	-	-	le	-	+	-
56	30	république	-	de	le	Cameroun	-	-	+	-	camerounais	-	le	-	+	-
57	30	république	-	de	le	Cap-Vert	-	-	+	-	cap-verdien	-	le	-	+	-
102	30	république	-	-	-	Centrafrique	-	-	-	-	centrafricaine	+	le	-	+	-
58	30	république	-	de	le	Chili	-	-	+	-	chilien	-	le	-	+	-
108	30	république	populaire	de	-	Chine	-	+	+	-	chinoise	-	la	-	+	-
12	30	république	-	de	-	Chypre	-	-	-	+	chypriote	-	<E>	-	+	-
13	30	république	-	de	-	Colombie	-	-	-	+	colombien	+	la	-	+	-
59	30	république	-	de	le	Congo	-	-	+	-	congolais	-	le	-	+	-
119	30	république	démocratique	de	le	Congo	+	-	+	-	-	-	le	-	+	Zaïre
14	30	république	-	de	-	Corée	-	-	-	+	sud-coréen	-	la	-	-	Corée du Sud

Table 16 : échantillon de la table NNpr-république

On peut noter que *:Vietnam* fait appel à un graphe du même nom qui décrit toutes les variantes orthographiques.

4.3.7.7 Vers une reconnaissance automatique de groupes nominaux

Le lexique accumulé jusqu'à présent est de taille modeste : au total, nous avons codé manuellement environ 650 entrées, réparties en 50 tables. Nous avons d'abord repris les résultats de O. Piton et al. (1997) pour les noms de pays et constitué une liste des noms officiels des pays à l'aide de la liste diffusée par la Délégation Française à la langue française[88]. Le travail réalisé sur les noms d'îles par M. Garrigues (1995) nous a permis de répertorier les contraintes syntaxiques auxquelles ils sont soumis. Nous avons également listé tous les départements français, les régions françaises, les états américains et les provinces canadiennes. Par ailleurs, nous avons complété nos listes à l'aide de savants dosages d'autres types de lieu géophysiques : *mer*, *pic*, *vallée*, etc. Le lexique accumulé est donc voué à être largement complété. Cependant, d'un point de vue linguistique, ce petit ensemble nous a permis de mettre en lumière un vaste ensemble de phénomènes linguistiques facilement représentables dans des tables syntaxiques. Il est probable, dans le futur, que d'autres contraintes soient découvertes et ajoutées aux tables existantes. En fait, cette étude avait surtout pour but d'ouvrir une voie dans le traitement linguistique des noms propres de lieu et de montrer que la méthodologie du lexique-grammaire pouvait être appliquée avec succès à leur analyse syntaxique.

Un premier moyen de compléter automatiquement nos tables de manière spectaculaire, serait d'utiliser comme ressource le dictionnaire *Prolintex* qui contient pour certains types de noms

[88] http://www.culture.fr/culture/dglf/ressources/pays/FRANCAIS.HTM

de lieu (les villes par exemple) toutes les informations syntaxiques nécessaires. En effet, les noms de ville ont un comportement syntaxique stable : ils ont tous pour forme *la ville de Det Npr* (avec *Det = le, la, les, E*). La seule information nécessaire pour les coder est de connaître le déterminant associé au nom propre et cette information est codée dans *Prolintex* comme le montrent les les exemples ci-dessous :

Laval,Laval.N+PR+Top+Ville+DetZ:ms:fs
Pirée,Pirée.N+PR+Top+Ville+DetLe:ms

A chaque nom de ville est associé un certain nombre d'informations : la catégorie grammaticale (*N*), le type (nom propre : *+PR* ; toponyme : *+Top*), la classe locative (*+Ville*), un trait syntaxique (déterminant *+DetZ*, *+DetLe*, etc.) et des informations flexionnelles (*ms* pour masculin singulier). Pour ajouter dans nos tables les noms de villes, il suffit d'extraire dans *Prolintex* les entrées candidates : par exemple, à l'aide de l'expression régulière :

N\+PR\+Top\+Ville\+(DetZ+DetLe+DetLa+DetLes+DetL)[89]

Puis, connaissant le format de la table des noms de ville, on ajoute l'entrée en complétant automatiquement ses colonnes. Ce travail est difficilement réalisable par un linguiste car il faut des notions informatiques.

La première application du codage de nos tables consiste à reconnaître des groupes nominaux ayant pour tête un nom propre composé de lieu géographique. Il suffit de convertir les tables en graphes de la même manière qu'avec les expressions de mesure. Pour chaque table, il convient de construire un graphe patron (ou graphe paramétré). Mais, étant donné le nombre important de tables, une telle démarche serait fastidieuse. Une solution plus pratique est de construire automatiquement l'ensemble des graphes patrons à partir d'une table générique (ou méta-table) et d'un graphe générique (ou un méta-graphe patron), comme l'a fait S. Paumier (2003) pour transformer les tables syntaxiques des verbes du français en graphes. Dans la table générique, chaque ligne correspond à une des tables syntaxiques. Les colonnes correspondent à l'ensemble des propriétés syntaxiques répertoriées dans notre étude. Etant donné une ligne (soit une table), chaque colonne indique si la propriété associée à celle-ci est codée dans la table. Si c'est le cas, il suffit d'indiquer dans quelle colonne de la table sélectionnée se trouve la propriété : par exemple, si la propriété se trouve dans la colonne *D*, on inscrit *@D* dans la colonne correspondant à cette propriété dans la méta-table. Si cette propriété n'est pas codée dans la table, cela signifie que pour toutes les entrées de la table, cette propriété est implicitement vraie ou fausse. Si la propriété est vrai, on insère le signe +, sinon on insère le signe -. La méta-table contient donc des booléens ou des indices de colonnes des entrées (des tables).
Nous avons répertorié l'ensemble des propriétés. Puis pour chacune des tables, nous avons établi la correspondance entre ses propres propriétés et l'ensemble des propriétés de la méta-table. Nous donnons un extraît d'une méta-table de notre ensemble de tables de type *NNpr* (**Npr-île**, **Npr-département**, etc.). Nous n'avons pas répertorié toutes les propriétés afin d'obtenir une figure plus claire. Par exemple, la table **NNpr-republique** est décrite à la cinquième ligne. Si on la lit de gauche à droite, cette ligne indique que les formes longues sont implicitement au singulier (et pas au pluriel). Le classifieur *Nc* est codé dans la colonne C de **NNpr-republique**. Implicitement, on ne peut pas insérer de modifieur en position M2 (**la république surpeuplée de Croatie*), etc.

[89] Le symbole \+ reconnaît le caractère + ; le symbole + est l'opérateur d'union des expressions régulières ; le symbole * est l'opérateur de Kleene.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Table	Dctc singulier	Dctc Pluriel	Nc	Modif M2	Adj1	Prep	Dct (forme longue)	Npr	Variante Npr (forme longue)	LE Nc Adj1 de Dct Npr	LE Nc Adj1 de Npr	LE Nc Adj1 Npr	LE Nc de Dct Npr	LE Nc de Npr	LE Nc Npr	Npr-a	LE Nc Npr-a	(Dctc Nprc+Dct Npr) être un Nc	LE archipel de Dct Npr	Dct (forme courte)	Variante Dct (forme courte)	Dct Npr	Variante Npr (forme courte)
NNpr-ile	I@B	@B	@D	@E	-	@F	@G	@H	-	-	-	-	+	@I	@J	@L	@M	+	@K	@N	-	@O	-
NNpr-departement	+	-	@C	+	-	@D	@E	@F	-	-	-	-	+	@G	-	@H	@I	+	-	@E	-	@J	-
NNpr-mer	+	-	@C	@D	-	@E	@F	@G	@H	-	-	-	+	@I	@J	-	-	@K	-	@L	-	@M	-
NNpr-ocean	+	-	@C	-	@D	-	-	@E	-	-	-	+	-	-	+	-	-	+	-	@F	-	@G	-
NNpr-republique	+	-	@C	-	@D	@E	@F	@G	-	@H	@I	-	@J	@K	-	@L	@M	+	-	@N	@O	@P	@Q
NNpr-region	+	-	@C	+	-	@D	@E	@F	-	-	-	-	+	@G	@H	@I	@J	+	-	@K	-	@L	-
NNpr-lac	+	-	@C	+	-	@D	@E	@F	-	-	-	-	+	@G	@H	-	-	+	-	@I	-	@J	-

Table 17 : méta-table NNpr

Nous construisons ensuite le graphe générique qui est le graphe paramétré de toutes les tables, avec toutes les propriétés décrites dans les colonnes B à X de la méta-table. Nous donnons ce graphe ci-dessous. Chaque variable *@i* où *i* est l'identifiant d'une colonne de la méta-table fait référence à la propriété correspondante. L'application du méta-graphe patron à la méta-table génère les graphes patrons de nos tables de type NNpr. Pour chaque ligne, un graphe dont le nom est l'élément de la colonne A, est généré. On utilise la méthode de conversion montrée dans le chapitre sur les expressions de mesure. Cette méthode élimine les chemins représentant des propriétés non acceptées par l'entrée (signe -), garde ceux qui décrivent des propriétés autorisées et ajoute les informations que l'on utilise dans les propriétés. Par exemple, pour la première ligne, le graphe patron associé à la table **NNpr-île** sera automatiquement généré en élaguant et en ajoutant les informations nécessaires à une copie du méta-graphe patron. La variable *@I* sera remplacée par l'information '*@H*' qui, dans le graphe patron associé à **NNpr-île** identifie la colonne *Npr* de **NNpr-île**. La variable @L (qui symbolise la structure *LE Nc Adj de Npr*) sera supprimée du graphe patron. @N (qui symbolise la structure *LE Nc de Npr*) sera, quant à elle, remplacée par l'étiquette vide, ce qui autorisera automatiquement l'utilisation de la structure associée à cette variable. Pour cette entrée, nous obtenons le graphe patron ci-dessous (sous le méta-graphe patron) :

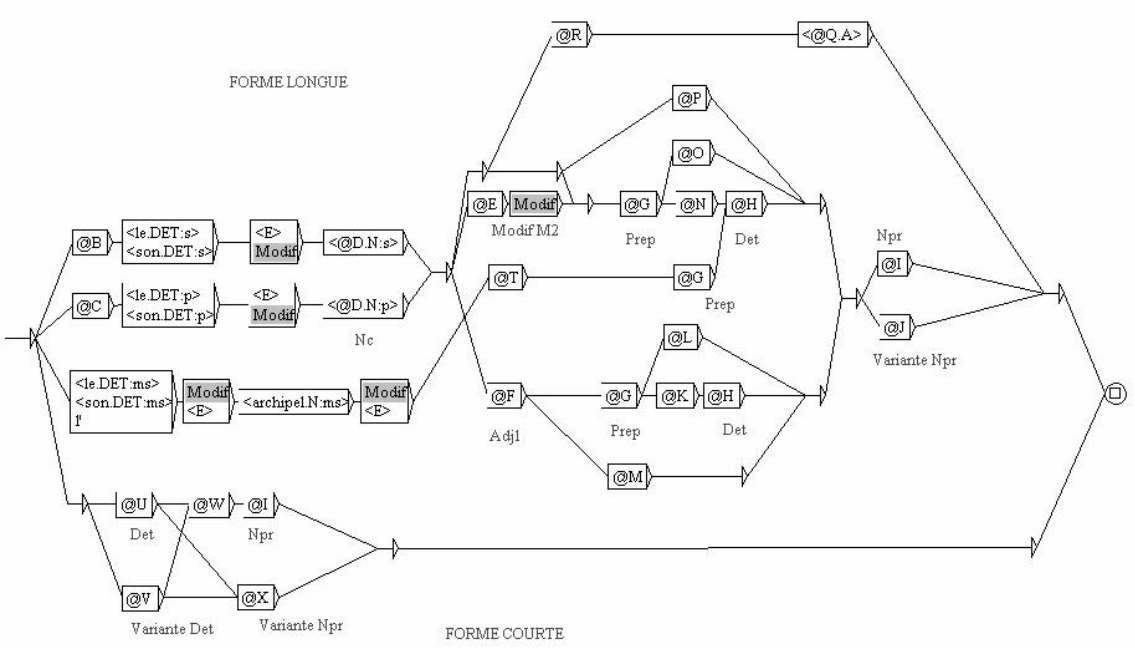

Figure 76 : méta graphe patron

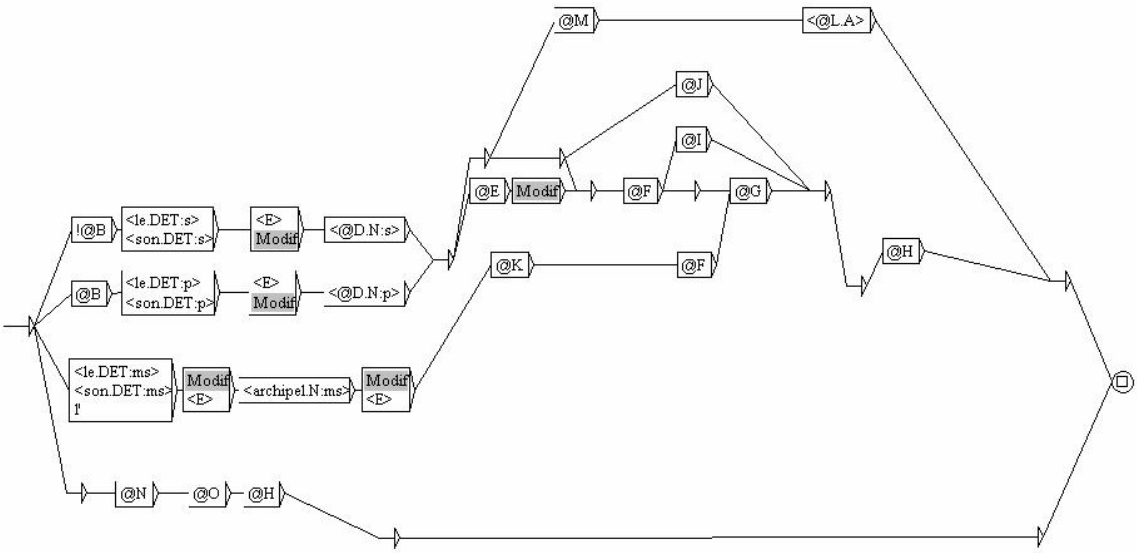

Figure 77 : graphe patron de la table NNpr-île

Le graphe fait référence aux colonnes par l'intermédiaire des variables *@B*, *@D*, *@E*, … correspondant respectivement aux colonnes *B*, *D*, *E*, … de la table **NNpr-île**. Le symbole *!@B* est la négation logique de *@B*. Ce symbole (utilisé dans Unitex) permet de décrire de manière simple une instruction du type *if … then … else…*. En effet, la colonne *B* indique si le classifieur est au pluriel (signe +) ou au singulier (signe -). Ainsi, le graphe patron doit contenir les deux possibilités. Les symboles (*@B* et *!@B*) permettent de sélectionner une des deux. L'étiquette *<@D.N:s>* indique que le classifieur sous forme lemmatisée, symbolisé par *@D* est un nom au singulier. *<@D.N:p>* indique qu'il est au pluriel. Le sous-graphe **Modif** est un graphe décrivant un modifieur adjectival quelconque, équivalent à l'expression régulière (<*ADV*> + <*E*>) <*A*> avec <*A*> adjectif et <*ADV*>

adverbe. En résolvant les références du graphe patron ci-dessus à la table **NNpr-île** pour l'entrée *île de Bornéo*, on obtient par exemple, le graphe ci-dessous.

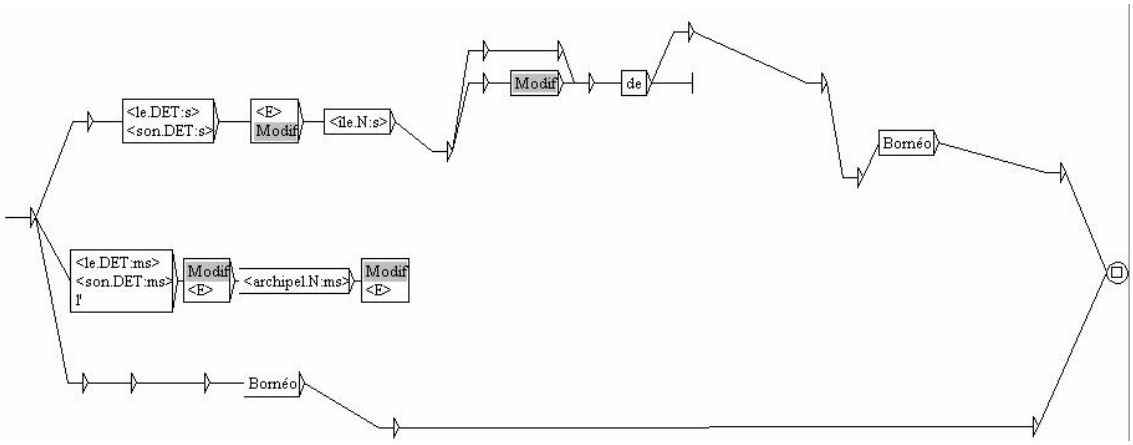

Figure 78 : entrée *île de Bornéo*

A partir de chacun des graphes patrons générés, nous générons les graphes associées à leurs entrées lexicales. Après union de ces graphes, nous obtenons un automate fini. Cet automate est en quelque sorte un mini-dictionnaire syntaxique de groupes nominaux simples. Nous avons confronté nos résultats à des textes en appliquant cet automate à une année du journal *Le Monde* (1994). Bien que nous ne puissions obtenir des résultats satisfaisant du fait de la non-exhaustivité de nos lexiques, ce test permet surtout de mettre en évidence les erreurs de codage : par exemple, une restriction trop forte ou simplement un oubli ou une erreur d'étourderie, etc. Par ce moyen, nous avons pu corriger un certain nombre de fautes.

Cette méthode de reconnaissance des groupes nominaux par transformation des tables en graphes pose des problèmes pour certains types de phénomènes linguistiques comme les coordonnées :

les villes de Paris et de Lyon
les états de Californie et du Texas

Ces expressions ne peuvent pas être représentées à l'aide de notre méthode car elles mettent en jeu différentes entrées lexicales dans un même graphe (supposé correspondre à une seule entrée). Ce problème est résolu si l'on change de stratégie en agissant en deux étapes :

(a) on transforme les tables en dictionnaire (type *Prolintex*)
(b) on construit des grammaires à l'aide de graphes utilisant les informations codées dans le dictionnaire

Par exemple, si l'on veut construire des grammaires reconnaissant des groupes nominaux avec des noms de villes coordonnés, il suffit de construire le graphe ci-dessous où l'on combine chaque classe de noms de villes qui prend un déterminant donné *Det* (ex : <N+Top+Pvil+DetLa>) avec ce même déterminant *Det* (ex : *la*). Il est évident que les noms de villes ont un graphe simple du fait de leur comportement syntaxique homogène. Pour les autres classifieurs, les grammaires construites deviennent rapidement complexes.

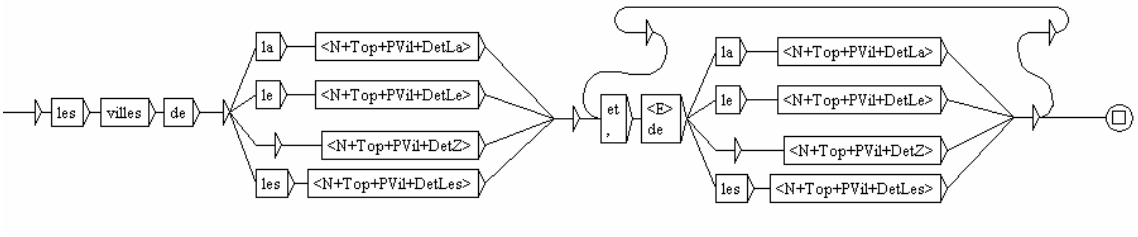

Figure 79 : coordination de villes

La difficulté de cette approche réside dans l'étape (a) que nous n'avons pas implémentée. Pour chaque ligne d'une table, il faut générer une entrée de dictionnaire contenant les informations suivantes : une forme fléchie, forme canonique, catégorie, traits syntaxiques et sémantiques [nom propre, locatif, classe de lieu, déterminants, etc.], informations flexionnelles. Les formes fléchies et canoniques correspondent à *Npr*, il suffit donc d'indiquer dans quelle colonne il est codé. La catégorie est clairement un nom (*N*). La difficulté réside dans l'extraction automatique des traits syntaxiques et lexicaux des tables pour les insérer dans l'entrée du dictionnaire. Il faut d'abord associer un nom compact, cohérent et unique à chacun de ces traits, puis trouver des procédures d'extraction. Comme ce type de travail n'est pas facilement accessible aux linguistes, il est nécessaire de prédéfinir des procédures d'extraction et donc de normaliser le codage des tables, ce qui n'entre pas dans le cadre de notre travail. Par ailleurs, pour implémenter cette approche, il faudrait aussi coder des informations flexionnelles dans les tables. Le gros inconvénient de cette approche est que les entrées du dictionnaire risquent de devenir illisibles du fait du grand nombre de propriétés. Par ailleurs, le phénomène de la coordination est général. Il existe le même problème avec les verbes par exemple :

Marie chante et danse.

Comme la représentation par grammaires des phrases simples au sein du lexique-grammaire est toujours en cours et loin d'être achevée, le traitement des coordonnées n'est pas d'actualité.

4.4 Description de groupes prépositionnels

4.4.1 Formes longues et variation prépositionnelle

Nous étudions la distribution prépositionnelle des noms propres composés *Nprc* = (*Npr*,*Nc*) qui entrent dans la construction à verbe support :

(Que P + N0) Vsup Loc Detc Nprc

Nous limitons notre ensemble des verbes supports au verbe *être*. Le verbe support *être* est un verbe neutre statique. Parfois, pour améliorer certaines acceptabilités, nous utilisons à la place les verbes *se passer*, *se dérouler* ou *se trouver* ayant un sens plus marqué. De même, nous limitons l'ensemble des prépositions *Loc* aux prépositions *à*, *dans*, *en* et *E* (préposition vide). Plus exactement, nous regardons les constructions suivantes :

(Que P + N0) être ((à + dans) le + en + E) Nprc

Nous ne regardons pas la préposition *sur* car son interprétation sémantique dépend trop du classifieur utilisé et du contexte.

Contrairement aux groupes nominaux, le comportement syntaxique de ces constructions ne dépend pas des deux éléments du couple (*Npr, Nc*), mais uniquement du classifieur *Nc*. Cela ne parait pas illogique car la nature et la forme du lieu sont explicitées par *Nc*. Par contre, là encore, un examen systématique est nécessaire car le comportement distributionnel des classifieurs est difficilement prédictible :

*Max est (dans l' + *à l + *en + *E) état de (le Texas + la Californie + Oregon)*
*Luc est (dans la + ?à la + en + *E) mer (du Nord + Noire+ Méditerranée)*
*Marie est (dans la + *à la + ?* en + *E) ville de (Paris + Le Havre + La Havane)*
*Léa est (dans la + à la + *en + E) rue (de la Paix + Monge)*
*Le randonneur est (?dans le + à le + *en + *E) pic (du Midi + du Vignemale)*

Il existe pourtant un cas particulier avec le classifieur *île* (M. Garrigues, 1995). En effet, l'emploi des prépositions *à et dans* n'est pas très naturel avec (*Corse, île*) et (*Sardaigne, île*) alors qu'il l'est avec les autres couples :

? Max est (à + dans) l'île de (Corse + Sardaigne)
Max est (à + dans) l'île de (Crête + Ré + la Martinique + la Guadeloupe)
Max est (à + dans) l'île Maurice

Cependant, l'interdiction n'est pas nette et nous considérons tous ces cas acceptables.

Le nom *région* a un comportement très spécifique. En effet, la préposition *en* ne peut s'employer avec toutes les structures nominales. Les structures nominales comprenant la préposition *de* ne peuvent pas se combiner avec cette préposition locative, alors qu'elles autorisent la préposition *dans* :

Léa est en région (Nord-Pas-de-Calais + Midi-Pyrénées + Ile-de-France)
** Léa est en région de (le Nord-Pas-de-Calais + le Midi-Pyrénées + Ile-de-France)*

Cette restriction n'est pas valable pour les autres classifieurs acceptant la préposition *en* :

Luc est en vallée d'Aspe
Luc est en mer de Corée

Concrètement, nous avons extrait de nos tables de noms propres tous les classifeurs : nous en comptons environ 70[90] en ajoutant quelques variations lexicales[91]. Pour chacun d'entre eux, nous avons systématiquement regardé et codé sa distribution prépositionnelle dans la table syntaxique **PNNpr** dont nous donnons un échantillon ci-dessous :

[90] A terme, le nombre de classifieurs va augmenter au fur et à mesure que le nombre de noms propres augmentera.
[91] Par exemple, la classe des villes est aussi sélectionnée par les classifieurs *village, commune, station balnéaire*, etc.

Loc Detc Nprc							
Loc = :à	Loc =: dans	Loc =:sur	Loc =: en	Loc =:E	Nc	Index Nc	en Nc...de ...
+	+	+	-	-	aiguille	1	-
-	+	+	-	-	archipel	2	-
+	+	+	-	+	avenue	3	-
-	+	+	+	-	baie	43	+
+	-	+	-	+	boulevard	68	-
-	+	+	-	-	bourg	44	-
-	+	+	-	-	bourgade	67	-
+	-	+	-	-	butte	45	-
-	+	+	-	-	canton	46	-
+	+	+	-	-	cité	47	-
-	+	+	-	-	cité	48	-
+	+	+	-	-	col	4	-
-	-	+	-	-	colline	49	-
+	+	+	-	-	Commonwealth	5	-
-	+	+	-	-	commune	50	-
-	+	+	-	-	comté	51	-
-	+	+	-	-	confédération	6	-
-	+	+	-	-	Cordillère	7	-
+	+	+	-	-	côte	52	-
-	-	+	-	-	côte	53	-
-	+	+	-	-	département	8	-
-	+	+	-	-	département d'outre-mer	71	-
-	+	+	-	-	désert	9	-
+	+	+	-	-	émirat	10	-
-	+	+	-	-	étang	54	-
-	+	+	-	-	état	11	-
-	+	+	-	-	état	69	-
+	+	+	-	+	États-unis	12	-
-	-	+	-	-	étoile	13	-
-	+	+	-	-	fédération	14	-
-	+	+	-	-	fleuve	15	-
-	+	+	-	-	gave	55	-
-	+	+	-	-	ghetto	56	-
+	+	+	-	-	glacier	57	-
-	+	+	-	-	golfe	58	-
+	+	+	-	-	grand-duché	16	-
+	+	+	-	-	île	17	-
+	+	+	-	-	impasse	59	-
+	+	+	-	-	lac	18	-

Table 18 : échantillon de la table PNNpr

On remarquera que *Commonwealth* et *Etats-Unis* sont des classifieurs car on a :

Le Commonwealth de (les Bahamas + la Dominique)
Les Etats-Unis de (Amérique + le Mexique)

La préposition *à*[92] pose souvent des problèmes d'acceptabilité car elle est très fréquemment utilisée dans le discours populaire et il est difficile de trouver une limite entre ce qui est acceptable ou pas. Dans notre cas, les phrases suivantes sont peu naturelles :

? Le rendez-vous est à la rue (Monge + de la Paix)
? Le rendez-vous est à l'avenue des Champs-Élysées

[92] La préposition *à* est un gros sujet d'étude (cf. B. Lamiroy, 2002).

Nous sommes confrontés à un autre cas litigieux lorsque la préposition *à* est combinée avec le classifieur *côte* (au sens de pente). En effet, la phrase suivante parait peu naturelle sans contexte :

? Le cycliste est à la côte Saint-Martin

Elle l'est plus si la côte Saint-Martin est considérée comme un point parmi un itinéraire de difficultés pré-établies.

On peut noter, d'autre part, que la préposition *dans* est très peu naturelle avec le classifieur *avenue* ou *boulevard*, alors qu'elle l'est avec le nom *rue*, alors qu'ils appartiennent tous trois à la même notion sémantique.

? La course a lieu dans l'avenue (des Champs-Elysées + Montmarte)*
? La course a lieu dans le boulevard Haussmann*
La course a lieu dans la rue Monge

Ces classifieurs sélectionnent préférentiellement la préposition *sur* :

Luc est sur l'avenue (des Champs-Elysées + Montmarte)
Luc est sur le boulevard Haussmann

L'étude de la distribution prépositionnelle des classifieurs permet de distinguer clairement plusieurs emplois ambigus. D'abord, le nom *cité* au sens de « quartier » se combine naturellement avec la préposition *à* alors que ce n'est pas vrai pour le nom *cité* au sens de « ville » :

Max (habite + se trouve) à la Cité des (Ulis + Sapins) (« quartier »)
** Luc (habite + se trouve) à la cité (E + médiévale) de Carcassonne* (« ville »)

De même, le nom *côte* au sens « pente » accepte sans difficulté la préposition *dans*, ce qui n'est pas le cas du nom *côte* au sens « bord de mer ».

Max (habite + se trouve) dans la côte Saint Martin (« pente »)
** Max (habite + se trouve) dans la côte d'Azur* (« bord de mer »)

Remarque :
Dans le cas hypothétique où la distribution prépositionnelle dépendrait des deux éléments du couple (*Npr*, *Nc*), il suffit de décrire ces couples séparément des autres dans des tables décrivant à la fois le comportement interne et la distribution prépositionnelle de la forme longue.

4.4.2 Formes courtes et variation prépositionnelle

Pour l'instant, nous avons étudié le comportement distributionnel des formes longues des noms propres dans des groupes prépositionnels. Il est intéressant de le comparer avec celui de leurs formes courtes associées. Les formes courtes sont des formes réduites des formes longues. Il est donc logique que leur distributions prépositionnelles soient identiques comme dans :

La croisière se déroule (dans la + sur la + en) (mer + E) Méditerranée
Max est perdu dans (le désert de + E) le Sahara
La rencontre a lieu (dans + à + sur) les (îles + E) Maldives
Max est (?à + sur + ?dans) (l'avenue de + E) les Champs-Élysées
Le coureur cycliste se trouve (à + dans + sur) (le col de + E) le Tourmalet

Ce phénomène naturel et logique n'est pas une généralité. Il existe de nombreux exemples montrant une disparité dans les distributions entre les formes longues et leurs formes courtes associées, comme le montrent les quelques exemples ci-dessous :

*Paul est à (*la ville de + E) Paris*
*Marie est à (l'île de + *la + *E) Crète*
Léa est en (**état de + E*) *Californie*
*Max est en (principauté de + *E) Monaco*
Le président est (*avenue de les + *les + *E*) *Champs-Élysées*

Pour certains classifieurs comme *département, île, état, province*, etc., la séquence *dans la* a parfois tendance à se contracter dans la préposition *en*, mais seulement lorsque le classifieur est absent :

Luc est dans (le département de + E) la Meurthe-et-Moselle
Luc est en Meurthe-et-Moselle
** Luc est en département de la Meurthe-et-Moselle*

L'argent se trouve dans (l' île de + ?E) la Martinique
L'argent se trouve en Martinique
** L'argent se trouve en île de la Martinique*

Marie est dans (l'état de + ?la) Caroline du Nord
Marie est en Caroline du Nord
**Marie est en état de Caroline du Nord*

Mais, ce phénomène n'est pas une règle générale car, pour certains noms propres, la préposition *en* est interdite :

*L'argent se trouve dans (l' île de + ?*E) la Réunion*
** L'argent est en Réunion*

Luc est dans (le département de + ?E) la Côte d'Or
*?*Luc est en Côte D'or*

Cette contraction est également observée pour les noms propres *Npr* commençant par une voyelle :

Luc est dans (le département de + ?E) l'Isère
Luc est en Isère

Marie est dans (la province de + E) l' Alberta
Marie est en Alberta

Là encore, il existe des contre-exemples comme pour (*Yonne,département*) :

Luc est dans (le département de + E) l'Yonne
**Luc est en Yonne*

De même, du fait de la présence du nom *île* à l'intérieur de *Npr* la préposition *en* ne se combine pas avec le couple (*Ile-du-Prince-Edouard, province*).

Mon fils est retenu dans la province de l'Ile-du-Prince-Edouard
*Mon fils est retenu (à l'+*en) Ile-du-Prince-Edouard*

On notera, cependant, que *Ile-de-France* se combine avec la préposition *en* :

Mon fils est retenu en Ile-de-France

Par ailleurs, les noms propres *Npr* prenant pour déterminant *le* et ne commençant pas par une voyelle peuvent ne pas autoriser cette contraction :

Luc est dans (le département de + E) le Cher
**Luc est en Cher*

Léa est dans (l'état de + E) le Texas
**Léa est en Texas*

Cependant, ce n'est pas toujours vrai :

L'élection est dans (le département de + E) le (Loir-et-Cher + Val-de-Marne + Indre-et-Loire)
L'élection est en (Loir-et-Cher + Val-de-Marne + Indre-et-Loire)

Léa est dans (l'état de + E) le Wyoming
Léa est en Wyoming

Pour les noms de provinces canadiennes et d'états américains, on observe que la combinaison des formes courtes avec la préposition *à* est généralement possible lorsque le *Npr* prend *le* comme *Det* :

Cette insurrection populaire a eu lieu au (Texas + Wyoming) [*Nc =*: *état*]
Max se trouve au Québec [*Nc =*: *province*]

Pourtant, la combinaison entre *à* et *Vermont* (réduction d'*état du Vermont*), n'est pas acceptée :

** Cette insurrection populaire a eu lieu au Vermont* [*Nc =*: *état*]

Ainsi, bien que l'on constate certaines tendances générales pour chaque classifieur, ces quelques exemples montrent clairement que la distribution prépositionnelle des formes courtes des noms propres composés n'est pas toujours prédictible.

Les noms propres composés *Nprc* peuvent être divisés en deux catégories disjointes : ceux dont la distribution prépositionnelle des formes courtes dépend uniquement de leur classifieur (catégorie 1) et ceux dont la distribution prépositionnelle des formes courtes n'est pas prédictible à partir du classifieur (catégorie 2).
Nous reprenons chaque table de noms propres. Toutes les tables qui comportent des noms propres de la deuxième catégorie sont reprises. On y ajoute des colonnes représentant la distribution prépositionnelle de notre ensemble de prépositions locatives. C'est le cas pour la table **NNpr-île** pour laquelle nous ajoutons trois colonnes : la première correspondant à la préposition *en* (colonne P), la deuxième à la préposition *dans* (colonne Q) et la troisième à la préposition *à* (colonne R). En effet, le comportement prépositionnel des *Nprc* ayant le nom *île* pour classifieur est difficilement prédictible :

*Max est (*à la + en) (Crète + Corse)* (*Det =*: *la*)
*Max est (à la + *en) Réunion* (*Det =: la*)
Léa est (en + à) Haïti (*Det =*: *E*)
*Léa est (*en + à) (Bornéo + Maurice)* (*Det =*: *E*)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
Index Npr	Detc pluriel	Index Nc	Nc	Modif M2	Prep	Det	Npr	LE Nc de Npr	LE Nc Npr	LE archipel de Det Npr	Npr-a	LE Nc Npr-a	Det (forme courte)	Det Npr	en Npr	dans Det Npr	à Det Npr
1	+	17	île	-	-	-	Aléoutiennes	-	+	+	-	-	les	+	-	+	+
56	-	17	île	-	de	la	Ascension	-	-	+	-	-	la	-	-	-	-
2	+	17	île	-	-	-	Bahamas	-	+	+	bahamien	+	les	+	-	+	+
3	+	17	île	+	de	les	Baléares	-	+	+	-	-	les	+	-	+	+
57	-	17	île	+	de	la	Barbade	-	-	-	barbadien	+	la	+	-	-	+
34	-	17	île	-	de	-	Beauté	+	-	-	-	-	-	-	-	-	-
4	+	17	île	-	-	-	Bermudes	-	+	+	-	-	les	+	-	+	+
35	-	17	île	+	de	-	Bornéo	+	-	-	-	-	<E>	+	-	-	+
5	+	17	île	+	de	les	Canaries	-	+	+	-	-	les	+	-	+	+
58	+	17	île	+	de	le	Cap-Vert	-	-	-	cap-verdien	+	le	+	-	-	+
6	+	17	île	-	-	-	Caraïbes	-	+	+	-	-	les	+	-	+	+
7	+	17	île	-	-	-	Célèbes	-	+	+	-	-	les	+	-	+	+
36	-	17	île	+	de	-	Ceylan	+	-	-	-	-	<E>	+	-	-	+
37	-	17	île	+	de	-	Chypre	+	-	-	chypriote	+	<E>	+	-	-	+
38	-	17	île	+	de	-	Corse	+	-	-	corse	+	la	+	+	-	-
39	-	17	île	+	de	-	Crète	+	-	-	crétois	+	la	+	+	-	-
40	-	17	île	+	de	-	Cuba	+	-	-	cubain	+	<E>	+	-	-	+
59	-	17	île	+	de	le	Diable	-	-	-	-	-	le	-	-	-	-
41	-	17	île	+	de	-	Elbe	+	-	-	-	-	<E>	+	-	-	+
8	+	17	île	-	-	-	Féroé	-	+	-	-	-	les	+	-	+	+
9	+	17	île	-	-	-	Fidji	-	+	-	fidjien	+	les	+	-	-	+
60	-	17	île	+	de	la	Grenade	-	-	-	grenadien	+	la	+	-	-	-
61	-	17	île	+	de	la	Guadeloupe	-	-	-	guadeloupée	+	la	+	+	-	+
42	-	17	île	+	de	-	Guernesey	+	-	-	-	-	<E>	+	-	-	+
43	-	17	île	+	de	-	Haïti	+	-	-	haïtien	-	<E>	+	+	-	+
10	+	17	île	-	-	-	Hawaï	-	+	-	-	-	<E>	+	-	-	+

Table 19 : reprise de NNpr-île

Pour la table **NNpr-république**, nous ajoutons même une quatrième colonne représentant la séquence *à Npr* (différente de *à Det Npr*). Cette dernière colonne est nécessaire pour coder les propriétés du nom propre *république du Panama* car ce nom apparaît dans deux types de groupes prépositionnels :

Max se trouve à (E + le) Panama

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
Index Npr	Index Nc	Nc	Adj1	Prep	Det	Npr	LE Nc Adj1 de Det Npr	LE Nc Adj1 de Npr	LE Nc de Det Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det	Variante Det	Det Npr	Variante Npr	en Npr	dans Det Npr	à Det Npr	à Npr
106	30	république	-	-	-	-	-	-	-	-	dominicaine	+	-	-	-	-	-	-	-	-
124	30	république	socialiste	de	le	:Vietnam	+	-	+	-	vietnamien	-	le	-	+	-	-	-	+	-
1	30	république	-	de	-	Afrique de le Sud	-	-	-	+	sud-africain	+	la+l'	-	+	-	+	-	-	-
2	30	république	-	de	-	Albanie	-	-	-	+	albanais	+	l'+la	-	+	-	+	-	-	-
107	30	république	fédérale	de	-	Allemagne	-	+	+	-	allemand	-	l'+la	-	+	-	+	-	-	-
4	30	république	-	de	-	Angola	-	-	-	+	angolais	+	le+l'	-	+	-	+	-	-	-
95	30	république	-	-	-	Argentine	-	-	-	-	argentin	+	la+l'	-	+	-	+	-	-	-
5	30	république	-	de	-	Arménie	-	-	-	+	arménien	+	la+l'	-	+	-	+	-	-	-
6	30	république	-	de	-	Autriche	-	-	-	+	autrichien	+	la+l'	-	+	-	+	-	-	-
3	30	république	-	de	-	Azerbaïdjan	-	-	-	+	-	-	le+l'	-	+	-	+	-	-	-
117	30	république	populaire	de	le	Bangladesh	+	-	+	-	-	-	le	-	+	-	-	-	+	-
53	30	république	-	de	le	Bélarus+Belarus	-	-	+	-	-	-	le	-	+	-	-	+	+	-
7	30	république	-	de	-	Belau	-	-	-	+	-	-	<E>	-	+	-	-	-	+	-
51	30	république	-	de	le	Bénin	-	-	+	-	-	-	le	-	+	-	-	-	+	-
52	30	république	-	de	le	Bhoutan	-	-	+	-	-	-	le	-	+	-	-	-	+	-
8	30	république	-	de	-	Biélorussie	-	-	-	+	biélorusse	+	la	-	+	-	+	-	-	-
9	30	république	-	de	-	Bolivie	-	-	-	+	bolivien	+	la	-	+	-	+	-	-	-
10	30	république	-	de	-	Bosnie-Herzégovine	-	-	-	+	bosniaque	-	la	-	+	-	+	-	-	-
54	30	république	-	de	le	Botswana	-	-	+	-	-	-	le	-	+	-	-	-	+	-
118	30	république	fédérative	de	le	Brésil	+	-	+	-	brésilien	-	le	-	+	-	-	-	+	-
11	30	république	-	de	-	Bulgarie	-	-	-	+	bulgare	+	la	-	+	-	+	-	-	-
55	30	république	-	de	le	Burundi	-	-	+	-	-	-	le	-	+	-	-	-	+	-
56	30	république	-	de	le	Cameroun	-	-	+	-	camerounais	-	le	-	+	-	-	-	+	-
57	30	république	-	de	le	Cap-Vert	-	-	+	-	cap-verdien	-	le	-	+	-	-	-	+	-
102	30	république	-	-	-	Centrafrique	-	-	-	-	centrafricaine	+	le	-	+	-	-	+	+	-
58	30	république	-	de	le	Chili	-	-	+	-	chilien	-	le	-	+	-	-	-	+	-
108	30	république	populaire	de	-	Chine	-	+	+	-	chinoise	-	la	-	+	-	+	-	-	-
12	30	république	-	de	-	Chypre	-	-	-	+	chypriote	-	<E>	-	+	-	-	-	+	-
13	30	république	-	de	-	Colombie	-	-	-	+	colombien	+	la	-	+	-	+	-	-	-
59	30	république	-	de	le	Congo	-	-	+	-	congolais	-	le	-	+	-	-	-	+	-
119	30	république	démocratique	de	le	Congo	+	-	+	-	-	-	le	-	+	Zaïre	-	-	+	-
14	30	république	-	de	-	Corée	-	-	-	+	sud-coréen	-	la	-	-	Corée du Sud	+	-	-	-

Table 20 : reprise de NNpr-république

Les tables dont les noms propres font partie de la catégorie 1 ne sont pas modifiées. Les rares exceptions sont enlevées de la table et codées dans une table séparée. En l'état actuel des travaux, nous n'avons pas rencontré de tels cas. Nous ne codons pas dans la table **NNpr-ville** le fait que la préposition *à* ne soit acceptée qu'avec les formes courtes des noms de ville[93] : nous le ferons lors de la transformation des tables en graphes (cf. dernière section du chapitre).

4.4.3 Des adverbes figés locatifs

Les compléments prépositionnels examinés jusqu'à présent comprennent obligatoirement le constituant *Npr*. Mais que se passerait-il si l'on effaçait *Npr* ? Regardons la phrase ci-dessous :

> *Max est dans la vallée de la Maurienne*
> *= Max est dans la vallée*

Il peut s'agir d'un effacement avec coréférence :

> *Max habite dans la vallée de la Maurienne. Marie espère un jour lui rendre visite dans la vallée*

Cependant, intuitivement, *dans la vallée* peut aussi référer à une vallée non nommée, identifiable de façon unique grâce au contexte :

[93] Pour quelques noms de ville comme *Avignon* ou *Arles*, la présence de la préposition *en* est autorisée : *en Avignon*. Ce phénomène est difficilement explicable : peut-être le fait que ces villes étaient des états pontificaux ; certains parlent même de snobisme (notre source a souhaité garder l'anonymat).

Luc a skié toute la journée à Courchevel ; il va redescendre dans la vallée en voiture

Dans cette phrase, la vallée dont on parle est celle qui se trouve en contrebas de Courchevel. Ce phénomène est très courant comme le montrent les exemples ci-dessous :

Paul est dans la maison. Par la fenêtre, il voit que Paul est dans la rue

Devant cette difficulté d'interprétation sémantique, nous ne traitons pas ces cas et nous regardons plutôt les constructions figées locatives de la forme :

N0 être Loc Det Nc =: *Paul est en montagne*

Ces constructions ont été très peu étudiées dans le cadre des études sur les constructions *être Prep X* au sein du lexique-grammaire. La préposition *en* est corrélée à un fort degré de figement. Par exemple, elle ne peut pas se combiner avec les formes longues des noms propres de villes, alors que la séquence sans *Npr* est tout à fait naturelle :

Marie est en ville
*?*Marie est en ville de Paris*

Lorsque la forme longue accepte la préposition *en* comme pour les noms de mers, la séquence sans *Npr* a un sens plus générique :

Luc est en mer Méditerranée
Luc est en mer (sens générique)

Par ailleurs, la règle de « pseudo-effacement » du *Npr* est loin d'être régulière, c'est plutôt une exception, ce qui renforce le caractère figé de *en mer* :

Luc est en vallée d'Aspe
**Luc est en vallée*

Max est en baie des Anges
**Max est en baie*

La présence de la préposition *en* est naturelle devant le nom *région* (sans *Npr*), mais ce dernier est au pluriel :

Les ministres sont en régions

Dans cette dernière phrase, le nom *région* n'est pas au sens de région administrative (ex : *Aquitaine*) ou un nom de localisation autour d'un lieu (*la région de Lyon*) car la phrase signifie que les ministres se trouvent en province. D'ailleurs, si l'on prend le nom *province*, on observe un phénomène similaire :

Luc est en province

Cette phrase signifie que Luc est en France mais n'est pas à Paris. Le nom *province* n'a donc pas le sens d'une partie administrative d'un pays (*province du Nouveau-Brunswick*).

Notons que le nom *principauté* rentre dans un adverbe locatif utilisant la préposition *en :*

Luc est en principauté

Cette expression est une réduction de *en principauté de Monaco*.

Si l'on regarde le nom *cité* au sens de « quartier », il est relativement naturel de dire la phrase suivante :

Ma famille (habite + ?est) en cité
Où habite ma famille ? en cité

Par contre, on n'observe pas la même construction avec *cité* au sens de « ville » :

**Luc (habite + est) en cité*
Luc est en ville

Pour d'autres classifieurs, on observe également un figement avec la préposition *en*, mais l'emploi est non locatif car les phrases ne répondent pas à la question en *où* :

La course est en côte
Comment est la course ? en côte
*Où est la course ? *en côte*

Ce figement est quand même un moyen de différencier les deux emplois de *côte* car on ne peut avoir ce comportement avec le nom *côte* au sens de « bord de mer ».

Le classifieur *république* rentre aussi dans un adverbe figé (*en république*) qui n'a pas un sens locatif :

Nous sommes en république
*Où est-ce que nous sommes ? *en république*

Le nom *butte* entre dans une construction non locative semi figée du type *être Prep C Prep* où *C* est un nom figé avec la préposition :

Max est en butte à des problèmes
**Max est en butte*

Là encore, la construction ne répond pas à la question en *où* :

A quoi Max est-il en butte ? à des problèmes
*Où Max est-il en butte ? *à des problèmes*

Le figement est possible avec d'autres prépositions que *en*, comme *à*. Il existe par exemple plusieurs variantes figées ayant le même sens qui utilisent le classifieur *montagne* :

Marie est (en + à la) montagne

Nous décidons d'élargir le champ d'investigation à d'autres classifieurs de lieu tels que *campagne* qui est ambigu : *campagne* opposé à « ville » (sens locatif) et *campagne* comme « campagne électorale, publicitaire, etc. » (sens non locatif). Ces deux emplois se distinguent par leur distribution prépositionnelle : l'emploi locatif interdit la préposition *en* mais accepte la préposition *à* ; l'emploi non locatif ne se combine pas avec *être à* mais avec *être en*.

*Marie est (*en + à la) campagne*
*Luc est (en + *à la) campagne (E + électorale)*

Regardons maintenant le classifieur *territoire*. On observe, dans ce cas-là, un phénomène particulier : la préposition *en* n'est acceptée que si *territoire* est suivi d'un adjectif :

**Léa est en territoire*
Léa est en territoire (français + contaminé + ennemi)
**Léa est en territoire (de la France + qui appartient à la France)*

Par contre, l'utilisation de la préposition *sur* n'est pas contrainte même si certains cas sont limites :

Luc est sur le territoire (E + français + ?de la France + ?qui appartient à la France)

Le nom classifieur *banlieue* accepte aussi la présence de la préposition *en* au sein d'une expression figée :

Les troubles sont en banlieue

Ce nom peut aussi apparaître comme un nom de localisation spécialisé à la ville : dans la phrase ci-dessous, le sujet Luc est situé par rapport à *Paris*.

Luc est dans la banlieue (de Paris + parisienne)

L'emploi de la préposition *en* est autorisée même si elle est plus naturelle avec l'adjectif *parisienne* que le complément de nom *de Paris* :

Luc est en banlieue (?de Paris + parisienne)

Il est facile de faire le parallèle avec *région* dont nous avons montré précédemment qu'il pouvait être considéré comme un nom de localisation spatiale dans certains cas :

Luc est dans la région (de Lyon + lyonnaise)

L'emploi de la préposition *en* est difficile si le nom *région* n'est pas suivi d'un adjectif :

*Luc est en région (*de Paris + parisienne)*

Nous avons répertorié manuellement les adverbes figés locatifs. Ils pourraient très bien être intégrés à la table EPC de M. Gross, mais, pour une meilleure lisibilité du lecteur, nous les représentons dans un graphe car ils sont en petit nombre. Notons que nous avons inséré des expressions utilisant la préposition *dans* comme *dans les îles*, *dans la rue*, *dans le désert* ou *dans la région* car elles paraissent relativement figées.

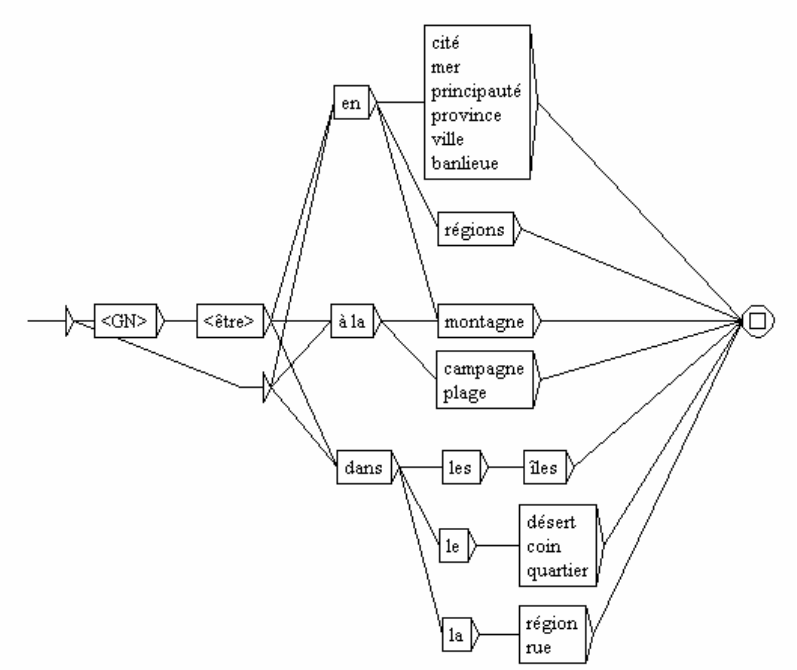

Figure 80 : EPC-locatif

Si l'on applique ce graphe (ou du moins la partie à droite du verbe *être*), on constate que le bruit est très important car le contexte d'analyse est très local. Nous donnons ci-dessous quelques exemples d'erreurs que l'on retrouve dans les textes :

- prépositions sélectionnées par le verbe de la phrase :

```
entreprises avaient contribué en 1993 à la campagne électorale.
party de adieu dans une cabine adossée à la montagne(…)
```

- analyse trop courte du groupe nominal

```
un séjour dans l'Ouest canadien, dans la région de Cariboo-Chilcotin,(…)
jeudi 17 mars à le petit matin, dans la région toulonnaise, ont été (…)
nombre de personnes inscrites à l'ANPE dans la région Ile-de-France.(…)
ne collision sur le Rhin, un naufrage en mer de le Nord et de les (…)
```

- autre :

```
les enseignants, à propos ?{S} De ville en ville et de chaîne en chaîne,(…)
```

4.5 Un nouveau système de conversion des tables en graphes

4.5.1 Préliminaires

4.5.1.1 Rappels sur la méthode standard de conversion

Nous rappelons brièvement la méthode standard de conversion. Comme nous l'avons déjà utilisée précédemment sous la forme d'exemples, nous la décrivons de manière formelle pour être cohérent avec la suite. L'objectif est de convertir une table en automates finis (usuellement appelés graphes dans la communauté RELEX) à l'aide d'un automate (ou graphe) de référence (ou automate paramétré). Formellement, un automate fini est un 4-uplet $<Q, I, F, \Sigma, \delta>$ où Q est un ensemble d'états, I est l'ensemble des états initiaux, F est l'ensemble des états finaux, Σ est l'alphabet et $\delta \subseteq Q \times \Sigma^* \times Q$ est l'ensemble des transitions. Pratiquement, les automates finis sont représentés à l'aide de graphes où les transitions sont des boites et les états ne sont pas représentés (sauf l'état initial et l'état final).

On suppose que Σ est un alphabet quelconque disjoint de l'alphabet des booléens ({+,-}). Soit M une table avec n colonnes qui contient des booléens et des éléments de Σ^*. Le graphe de référence associé à M est un automate fini dont l'alphabet est $\Sigma \cup \Delta$: Δ est un alphabet auxiliaire disjoint de Σ. Chaque élément de Δ est appelé variable et correspond à une colonne de M. Pratiquement, cet automate contient des variables $@j$ où l'entier $j \in [1, n]$[94] correspond à la colonne j de M. Le symbole *!* représente le NON logique et peut parfois être placé avant la variable.

Nous montrons maintenant l'utilisation de ce graphe de référence pour convertir la table M en graphes. Soit $T_i = < Q_i, I_i, F_i, \Sigma_i, \delta_i>$ une collection de graphes. T_0 est le graphe de référence (ou paramétré) associé à la table M. Pour tout $i > 0$, T_i sera le graphe associé à la ligne i de M et sera automatiquement construit à l'aide de l'algorithme suivant :

[94] Nous supposons ici que le symbole *j* est un entier pour des raisons de facilité. Nous avons vu dans les exemples que cela pouvait être un autre type d'identifiant : *A, B, …, Z, AA, AB,* etc.

```
pour chaque ligne i de M
        Ti ← T0.copie()
        pour chaque transition t = (q,a,q')∈ δi
                si a ∈ Δ
                // a = [ !]@j[95]
                        val ← M(i,j)
                        si a = !@j alors
                                si M(i,j) = + alors
                                        val ← -
                                finSi
                                si M(i,j) = - alors
                                        val ← +
                                finSi
                        finSi
                        si val = + alors
                                Ti.modifierEtiquette(t,ε)[96]
                        sinon
                                si val = - alors
                                        Ti.supprimerTransition(t)
                                sinon
                                        Ti.modifierEtiquette(t, val)
                                finSi
                        finSi
                finSi
        finPour
    finPour
```

Nous avons utilisé cette méthode pour convertir nos tables de groupes prépositionnels locatifs de manière approximative. L'approximation est dû au fait que nous ne tenons pas compte des tables des noms propres car nous réalisons une description générale de ces derniers dans le graphe-patron[97]. Le constituant *Npr* est décrit « graphiquement » dans le graphe **Npr** à l'aide des méta-étiquettes <*PRE*> et <*MOT*> qui désignent respectivement tout mot commençant par une majuscule et tout mot graphique (séquence de caractères). Le graphe-patron **PNNprApprox** représente l'ensemble des structures de la table **PNNpr** avec toutes les prépositions. Il est paramétré par les variables *@B*, *@C*, @D, @E qui correspondent respectivement aux prépositions *à* (colonne B), *dans* (colonne C), *en* (colonne D) et *E* (colonne E) et qui prennent une valeur lorsqu'on choisit une entrée de la table. La variable *@F* désigne le classifieur (colonne F). Pour chaque entrée de **PNNpr**, on peut alors construire automatiquement une version du graphe reconnaissant les groupes prépositionnels locatifs dans lesquels elle peut rentrer. Pour l'entrée *île* par exemple, les variables *@B* et *@C* sont remplacées par le mot vide <*E*>. Les boîtes contenant *@D* et *@E* sont supprimées. Enfin, *@F* est remplacée par l'entrée lexicale *île* se trouvant à la colonne F.

[95] [!] signifie que le symbole ! est optionnel.
[96] ε est le symbole vide ; la procédure *modifierEtiquette* modifie l'étiquette de la transition *t* de T_i par ε.
[97] Ceci est une reprise de M. Constant (2002).

A	B	C	D	E	F
Indcx Nc	Loc=: à	Loc =: dans	Loc =: cn	Loc =: E	Nc
1	-	+	-	-	département
2	+	+	+	-	mer
3	-	+	+	-	région
4	+	+	-	-	île
5	-	+	-	-	océan
6	-	+	-	-	état
7	-	+	-	-	état

Table 21 : table PNNpr

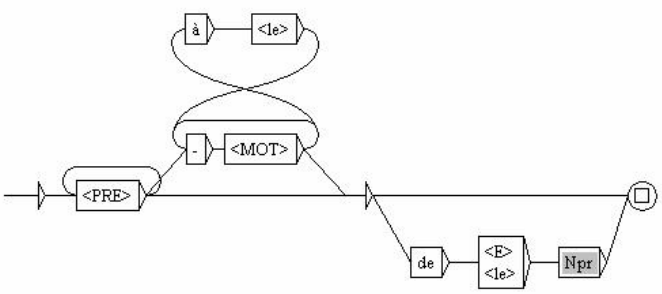

Figure 81 : Npr (générique)

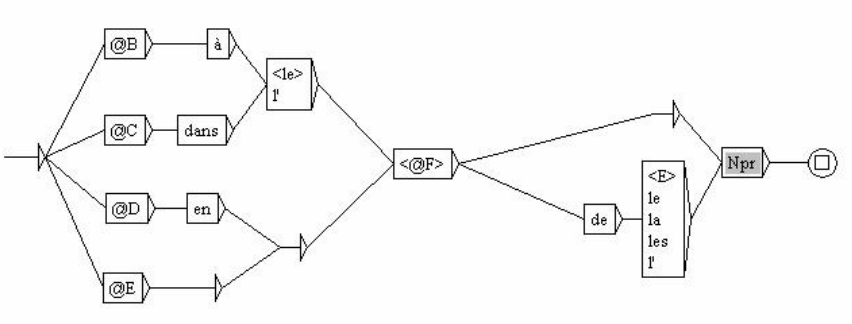

Figure 82 : graphe patron de PNNpr (approximation)

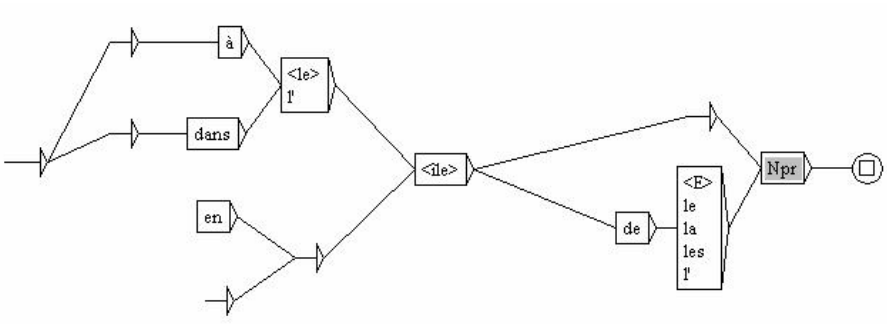

Figure 83 : entrée *île*

4.5.1.2 Un système relationnel

Dans la section précédente (cf. 4.4), nous avons construit un ensemble conséquent de tables. Les tables élaborées sont différentes de celles que l'on a l'habitude de construire. En effet, il

existe des relations entre elles : particulièrement entre la table des classifieurs et chacune des tables du type NNpr. Pour chaque table de noms propres, chaque ligne ayant pour classifieur *Nc* pointe sur la ligne de **PNNpr** qui décrit l'entrée de *Nc*. En fait, comme dans chaque table, tous les noms propres ont le même classifieur, toutes ses lignes pointent vers la même entrée de **PNNpr**. Mais cela est dû à notre choix de classification pour les noms propres. Si nous les avions classés selon leur structure interne, toutes les lignes de la table de type NNpr ne pointeraient pas sur la même ligne de **PNNpr** car il y aurait plusieurs classifieurs dans une même table. Ainsi, on peut dire que nous avons construit plusieurs systèmes relationnels de tables syntaxiques, composés chacun de deux tables (une table de type **NNpr** et la table **PNNpr**). Il est donc nécessaire de ré-examiner la méthode standard de conversion des tables en graphes, qui n'est plus valable dans ce type de système. Ce processus requiert des informations contenues dans les deus tables du système, ce qui n'est pas réalisable avec l'approche traditionnelle. Dans la suite, nous développons un modèle et un algorithme prenant en compte ces exigences.

Pour simplifier la compréhension du lecteur, nous supposons que la table de type NNpr contient différents classifieurs. Cet exemple sera utilisé tout au long de cette partie.

A	B	C	D	E	F	G	H	I	J
Indcx Npr	Indcx Nc	Nc	Prcp	Dct	Npr	LE Nc Prcp Dct Npr	LE Nc Prcp Npr	LE Nc Npr	Dct Npr
1	1	département	de	le	Nord	+	-	-	+
2	2	mer	de	le	Nord	+	-	-	-
3	3	région	de	le	Nord-Pas-de-Calais	+	-	+	+
4	1	département	de	l'	Oise	+	+	-	+
5	4	île	de	<E>	Oléron	-	+	-	+
6	5	océan	-	le	Pacifique	-	-	+	+
7	6	état	de	<E>	Israël	-	+	-	+
8	7	état	de	la	Californie	-	+	-	+
9	2	mer	-	la	Méditerranée	-	-	+	+
10	3	région	de	les	Pays de la Loire	+	-	-	+

Table 22 : NNpr

A	B	C	D	E	F
Indcx Nc	Loc=: à	Loc =: dans	Loc =: en	Loc =: E	Nc
1	-	+	-	-	département
2	+	+	+	-	mer
3	-	+	+	-	région
4	+	+	-	-	île
5	-	+	-	-	océan
6	-	+	-	-	état
7	-	+	-	-	état

Table 23 : PNNpr

Soit *M* une table qui contient des éléments lexicaux et des booléens (+ pour vrai et – pour faux). Chaque entrée lexicale est une clé primaire. La colonne contenant les entrées lexicales est appelée colonne primaire de la table. On appelle éléments secondaires les autres éléments lexicaux de la table : par exemple, dans **NNpr**, à la ligne 4, *département*, *de*, *le*, etc. sont des éléments secondaires alors que *Oise* est l'entrée lexicale. Le comportement syntaxique de certains éléments secondaires est parfois représenté indépendamment dans d'autres tables. Dans notre exemple, c'est le cas des *Nc* dont la distribution des prépositions est représentée dans la table **PNNpr**, lorsqu'ils sont associés aux noms propres *Npr*. Pour chaque ligne d'une table *M*, de tels éléments sélectionnent une ligne d'une autre table *M'*. Informellement, on peut considérer cela comme un appel à une sous-ligne qui contient les informations syntaxiques sur ces éléments. Par exemple, le classifieur *île* de l'entrée *île d'Oléron* de **NNpr** sélectionne la ligne 4 de **PNNpr** où *île* est la clé primaire. De tels éléments sont appelés clés secondaires. Les colonnes les contenant sont des colonnes secondaires. Une colonne clé est soit une colonne secondaire soit une colonne primaire.

Quelques détails techniques :

Notons qu'une clé est supposée unique dans la tradition des systèmes relationnels de bases de données (J.D. Ullman, 1979 ; G. Gardarin, 1999). Or, en réalité, nous avons vu que les entrées lexicales sont ambiguës. Par exemple, le nom *état* est ambigu dans la table **PNNpr** : c'est soit un classifieur de pays (*l'état d'Israël*), soit une partition administrative d'un pays (*l'état de Californie*)[98]. Dans la table **NNpr**, l'entrée lexicale *Nord* désigne soit une mer, soit un département français. Il est donc impossible que ces éléments utilisés tels quels soient des clés primaires. Pour régler ce problème, nous ajoutons, dans nos tables, une colonne dans laquelle, pour chaque entrée lexicale, nous associons un entier unique. Ainsi, chaque entrée a un identifiant unique qui permet de la différencier des autres : 6 est l'identifiant du classifieur *état* désignant un pays et 7 est celui correspondant au classifieur *état* désignant une partie administrative d'un pays.

4.5.2 Modélisation et algorithme

4.5.2.1 Un nouveau modèle

Notre système comprend un ensemble de n tables syntaxiques (M_1, M_2, M_3, …, M_n), un ensemble de relations entre elles et une table principale. On suppose désormais que M_1 est la table principale. Chaque table comporte une colonne primaire et peut comporter un certain nombre de colonnes secondaires[99]. Par exemple, dans notre système, nous avons deux tables syntaxiques reliées entre elles : M_1 = **NNpr** et M_2 = **PNNpr**. Leurs colonnes primaires sont les colonnes 1 des deux tables. La table principale est **NNpr**.
Une clé primaire d'une table M permet de référer directement à n'importe quelle ligne de M[100]. On adopte une numérotation absolue au sein du système pour chaque colonne clé de toutes les tables. Dans notre exemple, M_1 aura la colonne 1 (indice absolu K_1) pour colonne primaire et la colonne 2 (indice absolu K_2) pour colonne secondaire ; M_2 aura la colonne 1 (indice absolu K_3) pour colonne primaire. On a obligatoirement $K_1 \neq K_2$, $K_2 \neq K_3$ et $K_1 \neq K_3$. A chaque colonne secondaire K, on associe sa table cible. R est l'ensemble de tels couples. Dans notre exemple, les clés secondaires dans la colonne K_2 pointent sur les lignes de la table M_2 et ainsi, $R = \{(K_2,M_2)\}$.

Pour chaque ligne u de la table principale, il existe un automate (A_u) sur l'alphabet des indices (absolus) des colonnes secondaires (fig. 85). Un état correspond à une ligne d'une table et est représenté par un couple (r,m) où r est l'indice d'une ligne dans la table M_m. Soient $q = (rq,mq)$ et $p = (rp,mp)$ deux états de l'automate et K une colonne secondaire (qui est la colonne k dans la matrice M_{mq}). Une transition (q,K,p) signifie que l'élément de M_{mq} situé à l'intersection de la ligne rq et de la colonne k sélectionne la ligne rp dans M_{mp} (c'est-à-dire $(K,M_{mp}) \in R$ et $H_{mp}(M_{mq}(rq,k)) = rp$). L'automate A_u comporte un état initial qui correspond à la ligne u de la table principale M_1 (état $(u,1)$). Tous les états sont finaux. Par construction, comme les étiquettes sont des indices absolus (i.e. non relatifs à chaque table), l'automate est déterministe. Théoriquement, cet automate est susceptible de contenir des cycles. Nous donnons ci-dessous un exemple théorique avec quatre tables (M_1, M_2, M_3 et M_4) et cinq relations ($R=\{(K_1,M_4),(K_2,M_2),(K_3,M_2),(K_4,M_3),(K_5,M_3)\}$) ; v, w, x, y, z et s sont les indices de

[98] En général, dans le cas d'un classifieur de pays, *état* a une majuscule initiale (mais pas toujours).
[99] Une table peut ne pas comporter de colonnes secondaires.
[100] au moyen d'une table de hachage H_M qui associe un indice réel de ligne à chaque entrée lexicale (ou plutôt clé primaire). Par exemple, $H_{PNNpr}(2) = 3$ (équivalent de H_{PNNpr} (*mer*) = 3), car la première ligne réelle contient les intitulés des colonnes.

lignes sélectionnées par l'intermédiaire des relations à partir de la ligne *u* de la table principale. On fournit également l'automate associé à la ligne *u* de M_1. Si *q* est l'état (*v*,2) et *p* l'état (*x*,3), la transition (q,K_5,p) signifie que l'élément $M_2(v,k_5)$ sélectionne la ligne *x* de M_3 (k_5 est l'indice relatif dans M_2 correspondant à K_5).

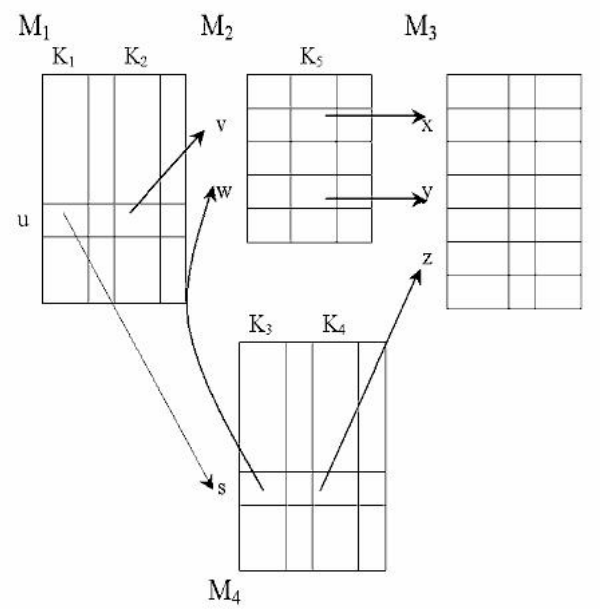

Figure 84 : système théorique

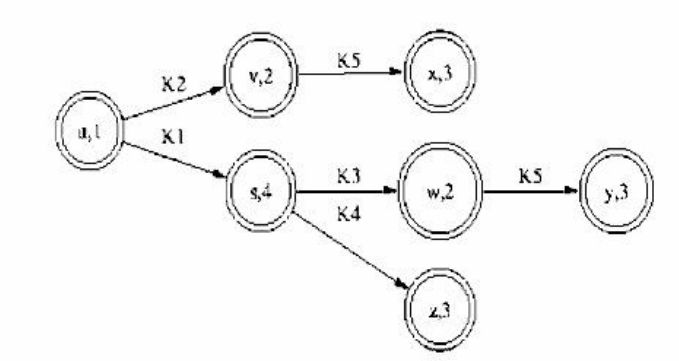

Figure 85 : automate *Au*

On dit que la ligne *v* de la table M_i est sélectionnée par la ligne *u* s'il existe un chemin dans l'automate partant de l'état initial à l'état (*v*,*i*). On dit aussi qu'une table est *m*-sélectionnée si elle contient *m* lignes sélectionnées.

4.5.2.2 Un graphe de référence avancé

Comme dans la méthode précédente de conversion, nous utilisons un graphe de référence (ou graphe paramétré). Le format des variables utilisées est différent car on traite un système avec *n* tables et non plus avec une unique table. Intuitivement, il est au moins nécessaire que les variables contiennent un indice de colonne et un indice de table. Comme les systèmes réels ne comprennent pas de relations complexes, impliquant des tables *m*-sélectionnées (avec $m > 1$), ces informations dans les variables devraient suffire dans la plupart des cas. Cependant, théoriquement, les graphes de référence requièrent plus d'informations dans les variables. C'est ce que nous allons montrer par la suite.

Comment atteindre un élément d'information ?

Pour chaque ligne u de M_1, nous souhaitons construire un graphe qui représente toutes les structures syntaxiques codées dans u et récursivement dans toutes ses sous-lignes. Chaque élément d'information se trouve dans un élément $M_i(v,j)$. Une méthode simple pour atteindre cet élément à partir de la ligne u de la table principale est d'avoir la séquence des colonnes secondaires successives utilisées pour sélectionner la ligne v. Si l'on applique l'automate déterministe A_u à cette séquence, l'état résultant est (v,i) correspondant à la ligne v de M_i. Ainsi, les variables du graphe de référence représentant des éléments d'information doivent contenir une séquence d'indices absolus de colonnes secondaires et la colonne désignant la propriété souhaitée. Comme le format de cette variable comporte une certaine complexité, pas forcément à la portée des linguistes, nous avons simplifié cette représentation.
En fait, la principale difficulté consiste surtout à déterminer une ligne parmi plusieurs lignes sélectionnées dans une table m-sélectionnée avec $m > 1$. Cependant, le cas $m > 1$ ne devrait pas être une situation très fréquente car les phénomènes linguistiques sont relativement simples. Etant donné cette remarque, nous proposons d'utiliser le format suivant pour les variables : $@i[:K_1:K_2:\ldots:K_l]:j$ où j est l'indice de la colonne contenant la propriété souhaitée ; i est l'indice de la table où l'information se trouve ; la séquence $K_1\,K_2\ldots\,K_l$ est la séquence des indices des colonnes secondaires pour atteindre la ligne sélectionnée dans la table i. Si M_i est 1-sélectionnée, $K_1\ldots K_l$ est optionnelle : en pré-calculant une table de hachage PH associant chaque table 1-sélectionnée à leur ligne sélectionnée, on accède directement à l'information. Autrement (si M_i est m-sélectionnée avec $m > 1$), la séquence de colonnes secondaires est obligatoire et le symbole i est redondant car il peut être déterminé en appliquant l'automate A_u à $K_1\ldots K_l$. Comme exemple, nous donnons le graphe de référence associé au système composé des deux tables M_1 (**NNpr**) et M_2 (**PNNpr**). Ces deux tables sont respectivement 0- et 1-sélectionnées. Les variables sont donc formées de deux paramètres (indice de table et indice de colonne). Exemple : la variable @1:6 correspond à *Npr* (colonne 6 de M_1) et la variable @2:3 correspond à l'acceptabilité de la préposition locative *dans* (colonne 3 dans M_2). Notons que nous n'utilisons plus les lettres majuscules (A,B, …,Z,AA,…) pour les indices des colonnes.

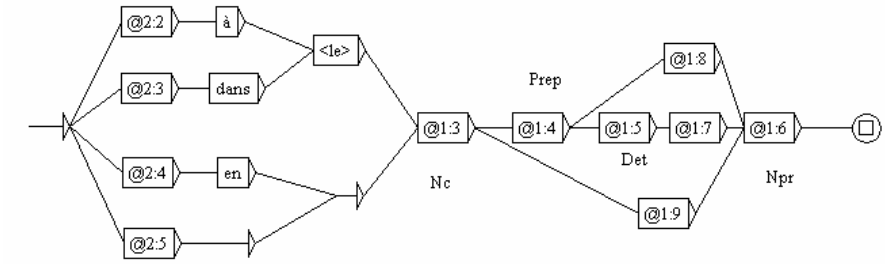

Figure 86 : graphe de référence évolué

4.5.2.3 Un nouvel algorithme

Nous proposons maintenant une extension de l'algorithme standard de conversion des tables syntaxiques en graphes. Soit $T_i = <Q_i, S_i, A_i, \Sigma\cup\Delta, \delta_i>$ une collection de graphes. T_0 est le graphe de référence. Pour $i > 0$, T_i est le graphe associé à la ligne i de M_1. L'algorithme suivant construit automatiquement T_u à partir de T_0 et le système de tables (Σ) :

```
Pour toute ligne u de M1
    Tu←T0.copie()
    Au←∑.constructionAutomate(u,M1)
    PH←∑.calculTableHachage(u,M1)
    Pour toute transition t = (q,a,q')∈ δu
        Si a ∈ Δ
            Si a = [!]@i:j //Mi est 1-sélectionnée
                v←PH(i)
            sinon //a= [!]@i:K1:… :Ki:j
                (v,i) ← Au.appliquerAutomate(K1…Kl)
            finSi
            val ← M(i,j)
            si négation logique dans a
                si M(i,j) = + alors
                    val ← -
                finSi
                si M(i,j) = - alors
                    val ← +
                finSi
            finSi
            si val = + alors
                Tu.modifierEtiquette(t,ε)
            Sinon
                Si val = - alors
                    Tu.supprimerTransition(t)
                Sinon
                    Tu.modifierEtiquette(t, val)
                FinSi
            FinSi
        finSi
    finPour
finPour
```

L'application de cet algorithme à notre système formé de deux tables fonctionne comme suit. Le programme commence à la première ligne de M_1 (entrée : *département du Nord*). Il réalise une copie du graphe de référence (T_0) et l'assigne à T_1. A_1 est automatiquement construit à partir des données générales sur le système entrées par le linguiste (tables, colonnes clés, relations). La table de hachage PH est ensuite construite. Chaque transition de T_1 est alors examinée et transformée si nécessaire (si l'étiquette contient une variable) :

- les transitions contenant @2:2, @2:4 et @2:5 sont supprimées car $M_2(1,2)$, $M_2(1,4)$ et $M_2(1,5)$ ont la valeur '-' ;
- les transitions contenant @1:8 et @1:9 sont également supprimées ;
- les étiquettes @2:3 et @1:7 sont remplacées par <E> car $M_2(1,3)$ et $M_1(1,7)$ ont la valeur '+' ;
- les étiquettes @1:3, @1:4, @1:5 et @1:6 sont respectivement remplacées par le nom *département*, la préposition *de*, le déterminant *le* et le nom propre *Nord*.

Ainsi, cette première étape du processus produit le graphe T_1 ci-dessous. Puis, le même processus continue pour les autres lignes de M_1.

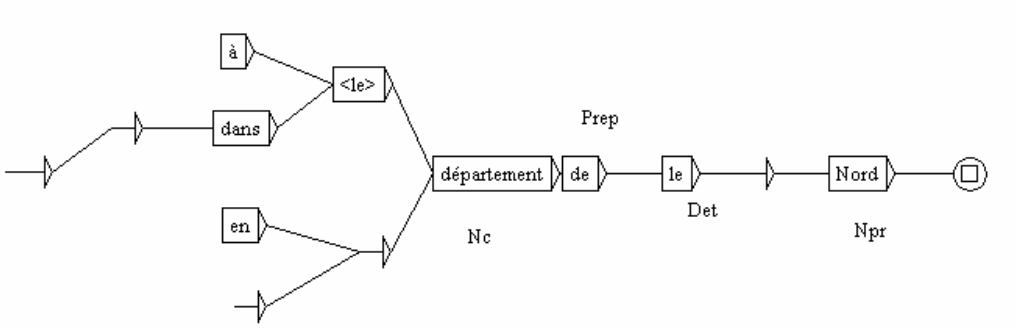

Figure 87 : entrée *département du Nord*

4.5.3 Application

L'analyse linguistique précédente (cf. 4.4) a conduit à la construction d'une table syntaxique de classifieurs représentant leur distribution prépositionnelle dans des groupes prépositionnels locatifs lorsqu'ils sont utilisés à l'intérieur de noms propres composés de lieu. Nous avons également élaboré des tables décrivant le comportement interne de différentes classes de noms propres. Nous obtenons un ensemble de systèmes relationnels. Nous proposons de convertir cet ensemble en graphes à l'aide d'une table générique (ou méta-table) et de la méthode mise en place ci-dessus. Comme précédemment, nous utilisons une table générique afin de générer le graphe paramétré pour chaque classe de noms propres. Pour construire cette table, nous reprenons la table générique élaborée pour décrire le comportement interne des noms propres. Les variables sont mises à jour : on transforme notamment les indices en entiers et l'on ajoute l'indice de la table. Par exemple, @C est remplacé en @1:3. On ajoute également des colonnes correspondant à la distribution prépositionnelle. Comme c'est indiqué dans la table ci-dessous, les noms propres dont la distribution de la forme courte dépend uniquement du classifieur, utilisent les informations codées dans la table 2 pour représenter leur comportement prépositionnel (ex : les noms de mer, de lac, etc.) ; pour les autres, ils utilisent les informations codées dans la table 1 (ex : les noms d'îles). Nous associons à cette table un méta-graphe paramétré au format standard (i.e. un seul paramètre par variable) décrivant l'ensemble des structures d'une entrée fictive qui rentre dans toutes les propriétés se trouvant dans les colonnes. Les variables correspondent aux propriétés de la table générique. Si l'on applique l'algorithme de conversion standard de la méta-table au méta-graphe paramétré, on obtient, pour chaque ligne (chaque table de noms propres), le graphe paramétré associé. Nous donnons ci-dessous le graphe de référence associé à la table **NNpr-île** généré à partir de ce processus automatique.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF
Table	Dets singulier	Dets Pluriel	Nc	Modif M2	Adj	Prep	Det (forme longue)	Npr	Variante Npr (forme longue)	LE Nc Adj Det Npr	LE Nc Adj de Npr	LE Nc Adj Npr	LE Nc de Det Npr	LE Nc de Npr	LE Nc Npr	Npr-a	LE Nc Npr-a	(Dets Npro+Det Npr) être un N	LE archipel de Det Npr	Det (forme courte)	Variante Det (forme courte)	Det Npr	Variante Npr (forme courte)	dans LE Npro	à LE Npro	en Npro	Npro	dans Det Npr	à Det Npr	en Npr	à Npr
MNpr-ile	I@1:2	@1:2	@1:4	@1:5	.	@1:6	@1:7	@1:8	.	.	.	.	*	@1:9	@1:10	@1:12	@1:13	*	@1:11	@1:14	.	@1:16	.	@2:2	@2:1	@2:4	@2:5	@1:12	@1:9	@1:7	.
MNpr-departeme	+	.	@1:3	+	.	@1:4	@1:5	@1:6	.	.	.	.	*	@1:7	.	@1:8	@1:9	*	-	@1:5	.	@1:10	.	@2:2	@2:1	@2:4	@2:5	@1:12	@1:11	@1:11	.
MNpr-mer	+	.	@1:3	@1:4	.	@1:5	@1:6	@1:7	@1:8	.	.	.	*	@1:9	@1:10	.	.	@1:11	.	@1:12	.	@1:13	.	@2:2	@2:1	@2:4	@2:5	@2:2	@2:1	@2:4	.
MNpr-ocean	+	.	@1:3	.	@1:4	.	@1:5	.	.	.	.	.	.	.	*	.	.	*	.	@1:6	.	@1:7	.	@2:2	@2:1	@2:4	@2:5	@2:2	@2:1	@2:4	.
MNpr-republique	+	.	@1:3	.	@1:4	@1:5	@1:6	@1:7	.	.	@1:8	@1:3	@1:10	@1:11	.	@1:12	@1:13	*	.	@1:14	@1:15	@1:16	@1:17	@1:13	@2:1	@2:4	@2:5	@1:5	@1:20	@1:18	@1:21
MNpr-region	+	.	@1:3	*	.	@1:4	@1:5	@1:6	.	.	.	.	*	@1:7	@1:8	@1:9	@1:10	*	.	@1:11	.	@1:12	.	@1:13	@2:1	@2:4	@2:5	@1:5	@1:16	@1:14	.
MNpr-lac	+	.	@1:3	*	.	@1:4	@1:5	@1:6	.	.	.	.	*	@1:7	@1:8	.	.	+	.	@1:9	.	@1:10	.	@2:2	@2:1	@2:4	@2:5	@2:2	@2:1	@2:4	.

Figure 88 : méta-table

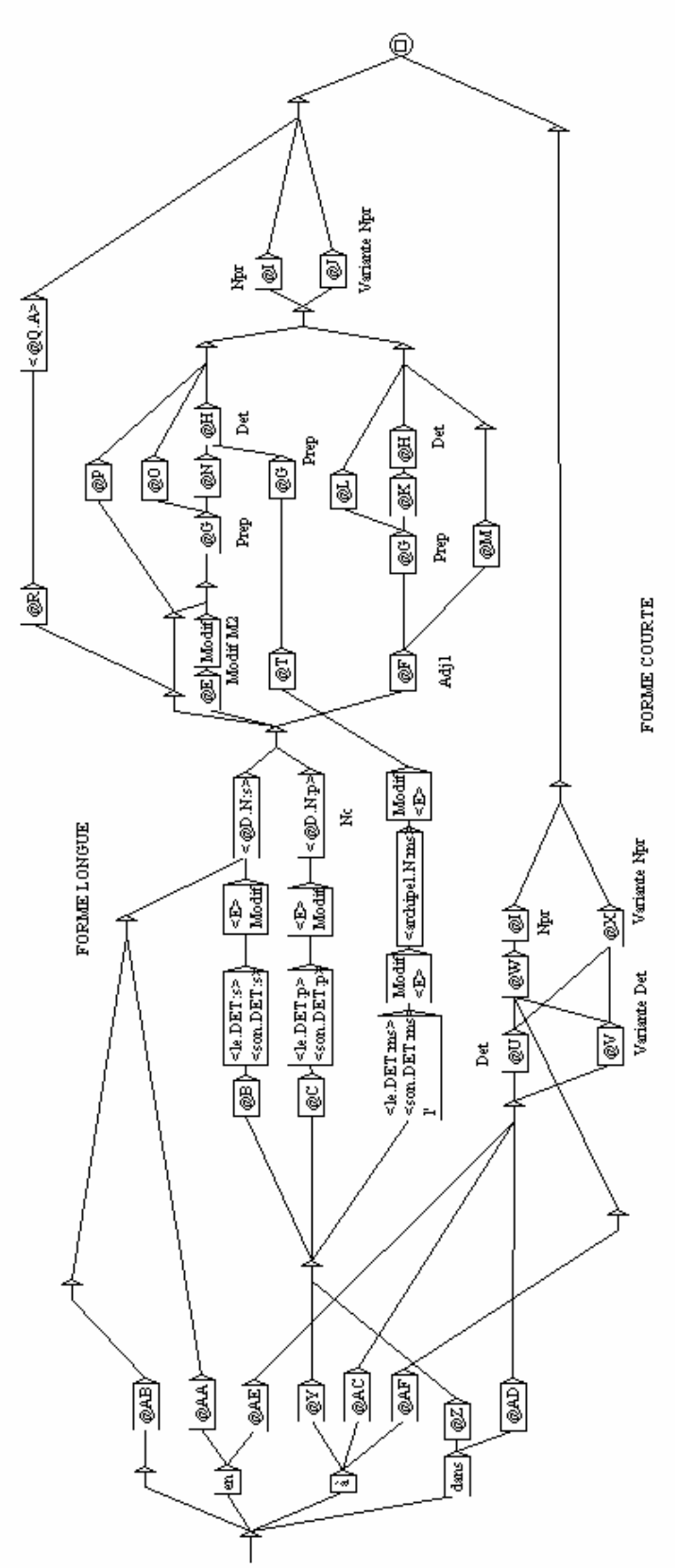

Figure 89 : méta-graphe paramétré PNNpr

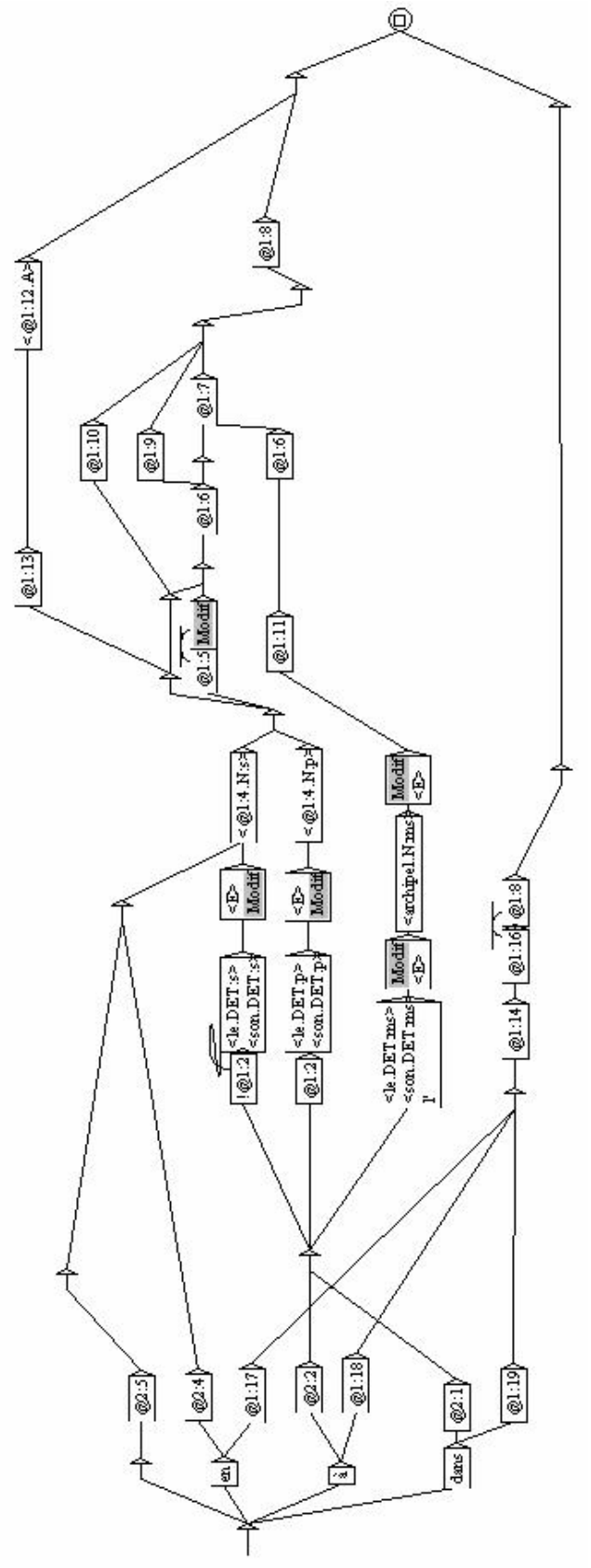

Figure 90 : graphe paramétré pour *NNpr-île*

Ensuite, nous construisons une autre table qui va nous permettre de définir chacun des systèmes relationnels. Chaque ligne de cette table correspond à un système relationnel. Chaque colonne contient une information sur le système. Les colonnes s'organisent consécutivement comme suit :

- nombre de tables du système
- pour chacune des tables : nom de la table, colonne primaire
- nombre de relations
- pour chacune des relations : table de départ, colonne secondaire de table de la table de départ, table cible
- nom du graphe patron associé au système

Ainsi, on construit manuellement la table suivante[101] :

2	NNpr-ile	1	PNNpr	7	1	1	3	2	PNNpr-ile
2	NNpr-departement	1	PNNpr	7	1	1	2	2	PNNpr-departement
2	NNpr-mer	1	PNNpr	7	1	1	2	2	PNNpr-mer
2	NNpr-republique	1	PNNpr	7	1	1	2	2	PNNpr-republique
2	NNpr-region	1	PNNpr	7	1	1	2	2	PNNpr-region

Table 24 : table de définition des systèmes relationnels

La première ligne signifie que le système comporte deux tables : M_1 = **NNpr-ile** (table principale) et M_2 = **PNNpr**. L'indice de la colonne primaire de la première est 1 et celui de la seconde est 7. Il existe une relation qui relie M_1 à M_2 par la colonne secondaire 3 de M_1. Le graphe patron associé à ce système est le graphe **PNNpr-ile** (voir ci-dessus).
Nous implémentons une procédure générale permettant d'extraire, pour chaque ligne, toutes les informations du système associé et puis appliquer le nouvel algorithme afin de générer les graphes de groupes prépositionnels locatifs pour chaque nom propre répertorié dans le système courant. Pour l'entrée lexicale *île de Bornéo* de **NNpr-ile**, nous obtenons le graphe suivant :

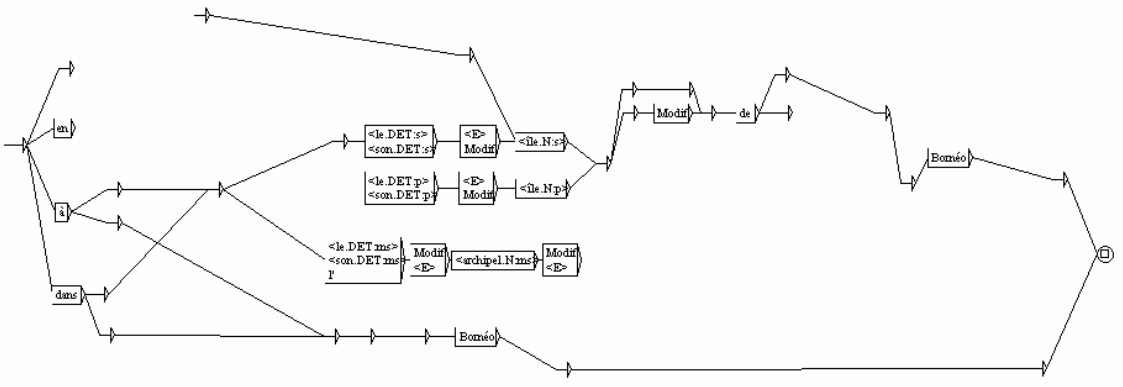

Figure 91 : *île de Bornéo* dans un adverbe de lieu

[101] Il est clair que, pour chaque ligne, le nombre de colonnes utiles varient selon le système défini (nombre de tables, nombre de relations).

4.6 Conclusion

L'analyse automatique des adverbes locatifs ne peut se résumer à un simple repérage des prépositions locatives et des groupes nominaux. L'étude des contraintes syntaxiques entre les constituants des adverbes est un premier moyen de régler le problème (M. Gross 1986, 1996). C'est ce que nous avons cherché à faire en nous attaquant à la distribution prépositionnelle des noms propres de lieu géographique au sein d'adverbes locatifs. Ce domaine n'est pas facile à traiter du fait de la spécificité des noms propres. Le travail que nous avons réalisé est une ébauche méthodologique dans le cadre du lexique-grammaire et mériterait d'être continué par des experts du domaine des lieux géographiques. Nous avons également mis au point une nouvelle méthode de conversion des tables en graphes s'appliquant à des systèmes de tables relationnelles (M. Constant, 2003). Cette représentation relationnelle ne se limite pas seulement à notre problème particulier (adverbes locatifs). En effet, les adjectifs avec noms appropriés semblent pouvoir être formalisés de cette manière (E. Laporte, 1995). Il en est de même pour certaines grammaires de dates (D. Maurel, communication personnelle).

Chapitre 5 : Un système de gestion de grammaires locales

5.1 Introduction

La quantité de ressources linguistiques augmente sensiblement. Ces ressources sont des composants linguistiques, parfois indissociables de certains outils linguistiques. Un des grands défis actuels et futurs est de construire des outils permettant de gérer ces ressources. Une des tendances est de mettre ces ressources à la disposition du public (ou de la communauté scientifique) sous la forme de logiciels libres ou de données librement accessibles via Internet :

- outils informatiques : Lingo [102] (formalisme HPSG), Xtag (TAG, Université de Pennsylvanie), Intex [103] et Unitex [104] (Dictionnaires morphologiques et tables syntaxiques, FSTs, RTN), AT&T's FSM[105], XRCE FS Tool[106], FSA Utilities[107] (transducteurs finis), etc.
- composants linguistiques : dictionnaires (Frantext, système DELA, etc.), réseaux sémantiques (Wordnet[108]), corpus (Frantext[109]), etc.

L'existence d'une association spécialisée dans la distribution de telles ressources (ELRA/ELDA) est une preuve supplémentaire. Des projets financés par l'État, comme Outilex (2002), ont pour objectif de rassembler et distribuer librement ces ressources. Certains chercheurs proposent des solutions innovantes. Par exemple, L. Romary (2000) décrit un système de serveur permettant de faire des liens entre différents serveurs contenant des ressources linguistiques (permettant en quelque sorte de rassembler ces ressources).

Nous nous intéressons précisément à la gestion des composants (ou données) linguistiques. Il est bien évident que leur éparpillement et la multiplication de projets de construction

[102] http://lingo.stanford.edu/index.html
[103] http://www.nyu.edu/pages/linguistics/intex/
[104] http://www-igm.univ-mlv.fr/~unitex/
[105] http://www.research.att.com/sw/tools/fsm/
[106] http://www.xrce.xerox.com/competencies/content-analysis/fssoft/docs/fst-97/xfst97.html
[107] http://odur.let.rug.nl/~vannoord/Fsa/
[108] http://www.cogsci.princeton.edu/~wn/
[109] http://zeus.inalf.fr/frantext.htm

provoquent inévitablement un problème d'hétérogénéité même s'il existe de nombreux projets de normalisation comme la standardisation des lexiques (EAGLES, 1996). Pour résumer, les trois grandes tendances actuelles sont la normalisation, la distribution libre et le rassemblement des données. Plus les données sont normalisées et libres, plus il est facile de les rassembler et les distribuer. Au sein du réseau RELEX, on observe également ces trois tendances :

(*Normalisation des données*)
Dès le début de la construction des données linguistiques, les notions fondamentales et les méthodes ont été communes à l'ensemble du réseau. L'utilisation des plates-formes Intex puis Unitex ont étendu cette standardisation aux formats utilisés dans les données.

(*Distribution libre*)
Unitex (S. Paumier, 2003) est totalement libre et open-source, les dictionnaires sont partiellement accessibles ; Intex (M. Silberztein, 1993) est gratuit pour les chercheurs.

(*Regroupement des données*)
Les dictionnaires sont construits par différentes équipes. Comme ils sont les fondements des logiciels, ils sont centralisés par les auteurs de ces derniers ; il existe un travail en cours pour les tables de lexique-grammaire.

Nous avons créé un prototype de système de gestion de grammaires locales [110]. La construction de telles grammaires est facile et rapide, si elle respecte une méthode rigoureuse et systématique (cf. précédents chapitres). Ainsi, leur nombre augmente de façon vertigineuse. De plus, leur formalisme permet leur réutilisation dans d'autres grammaires. Nous avons donc un ensemble de briques éparpillées au sein du réseau RELEX. Actuellement, il n'existe pas de gestion commune des grammaires, ce qui provoque parfois une certaine redondance dans les travaux. Les seuls moyens de s'informer ou d'informer de l'existence d'une grammaire sont les articles, les communications aux différents colloques ou les discussions orales ou écrites (courrier électronique). Une façon de remédier à ce problème est de centraliser les grammaires locales et de les mettre en libre accès via Internet aux chercheurs de la communauté. Ce travail est d'abord un travail d'ingénieur car il consiste à construire concrètement une bibliothèque de graphes en-ligne. Mais il a aussi un intérêt fondamental car nous traitons des objets qui ne sont pas habituellement utilisés dans de tels systèmes. Ceci implique la conception d'algorithmes de stockage (insertion et suppression) et de recherche d'information dans des objets complexes.

5.2 Spécifications et organisation du système

Le système que nous proposons est un système client-serveur distribué. Le serveur dispose d'une base de données relationnelle formée principalement de grammaires locales et autres informations (utilisateurs, etc.). Il comporte également un module traitant les requêtes arrivant du client. Le client est constitué d'une interface au moyen de laquelle l'utilisateur envoie ses requêtes au serveur. L'interface permet à l'utilisateur de visionner les résultats de ses requêtes. Les requêtes sont essentiellement de deux ordres : mise à jour des données et accès aux données.
Cette bibliothèque en-ligne où seront accumulées l'ensemble des grammaires locales servira de plate-forme d'échange de données pour la communauté RELEX. L'accès des utilisateurs au catalogue des grammaires locales existantes permettra d'éviter de la redondance dans les

[110] Nous traitons seulement des grammaires où l'unité minimale est le mot (et non le caractère).

travaux. Par ailleurs, le formalisme des grammaires locales encourage la réutilisation de grammaires déjà existantes : chaque chercheur peut incorporer dans ses graphes les briques des autres. Ainsi, il est nécessaire que chaque utilisateur puisse accéder à l'information et télécharger les graphes de la bibliothèque le plus facilement possible.

5.2.1 Spécifications

5.2.1.1 Mise à jour des données

Chaque utilisateur du système peut insérer de nouvelles grammaires dans la bibliothèque. Cette procédure est simple. Pour cela, le nombre de champs à remplir est limité afin de ne pas rendre la tâche pénible. Des procédures d'analyse automatique des grammaires ont également été mises en place (ex : calcul de la liste des graphes présents dans une grammaire donnée). Chaque utilisateur est responsable de ses propres grammaires, qu'il modifie ou supprime à sa discrétion. Les autres utilisateurs n'ont pas accès en écriture à ces graphes. Un utilisateur possède un compte personnel auquel il accède au moyen d'un nom d'utilisateur et d'un mot de passe. Il l'organise au moyen d'une arborescence de répertoires. Dans le futur, nous souhaiterions autoriser le fait que plusieurs utilisateurs puissent partager l'accès en écriture à un même graphe.

La politique adoptée pour le stockage des graphes est la suivante. L'utilisateur a le choix entre insérer ou supprimer un graphe seul (c'est-à-dire sans ses sous-graphes) ou une grammaire complète (un graphe principal et récursivement tous ses sous-graphes). Il est nécessaire de tenir compte de la dépendance entre les graphes : quel graphe utilise quel graphe ? Par exemple, on décide que l'on ne peut supprimer un graphe s'il est utilisé dans un graphe que l'on ne veut pas supprimer. Par ailleurs, nous avons une politique de duplication des graphes qui peut être coûteuse en espace mémoire mais qui nous semble indispensable. Supposons que la bibliothèque comporte un graphe G_A construit par un utilisateur U_1. Supposons maintenant qu'un utilisateur U_2 veuille insérer sur son compte un graphe G_B dont G_A est un sous-graphe. Une première solution consisterait à stocker physiquement G_B et l'appel à G_A serait symbolisé par un pointeur vers le graphe G_A dans le compte de U_1. Cette solution paraît à première vue idéale car elle permet une forte économie de mémoire. Cependant, selon le contexte de la grammaire locale, les deux graphes G_A peuvent évoluer différemment. Nous citons par exemple le cas des grammaires du discours boursier de T. Nakamura (à paraître) qui utilise les graphes généraux de dates de M. Gross mais les a adaptés à son contexte de travail. Le système de pointeurs n'autorise pas des évolutions divergentes des grammaires. Nous décidons donc de dupliquer les graphes du type G_A. Cette stratégie a plusieurs inconvénients. D'abord, elle est coûteuse en mémoire. Ensuite, elle supprime le lien qui existait entre les deux graphes de type G_A. Ainsi, lorsque l'un des deux est modifié, l'autre ne l'est pas automatiquement. Un bon moyen de résorber les effets néfastes de la duplication des graphes est d'avertir les auteurs lors de la modification d'un graphe qui les intéresse.

5.2.1.2 Accès aux données

L'un des buts de la bibliothèque est de donner aux utilisateurs l'accès au catalogue des grammaires locales stockées. Comme leur nombre risque d'exploser[111], il a été nécessaire d'implanter un moteur de recherche. Les critères de recherche sont multiples. Nous partons du plus simple au plus complexe. Tout d'abord, il est possible de construire un filtre sur des caractéristiques simples des grammaires tels que l'auteur, la langue, le type (avec ou sans sorties, avec ou sans variables, …), etc. Ce filtrage est une opération classique dans les bases de données relationnelles traditionnelles.

[111] Nous ne serions pas surpris de recueillir des dizaines de milliers de graphes.

Nous avons également implémenté une procédure de recherche de graphes à partir d'une description par mots-clés de la grammaire recherchée. Pour cela, une documentation des graphes (du moins, des plus importants d'entre eux) est nécessaire. Pour éviter de rebuter les auteurs, nous avons mis en place un éditeur de documentation. La procédure de recherche revient alors à une procédure classique de recherche documentaire dans une base de données textuelles (cf. moteurs de recherche Google, Yahoo, etc. : recherche de pages Web).
Enfin, on peut effectuer des recherches de grammaires à partir de leur contenu lexical. Un utilisateur peut donner un ensemble de mots ou étiquettes qu'il souhaite voir apparaître dans la grammaire qu'il cherche. Il peut également trouver toutes les grammaires de la bibliothèque qui contiennent ou reconnaissent les séquences qu'il désire.
La visualisation du résultat des recherches est fondamentale. Il convenait de trouver une représentation simple mais qui contient des informations nécessaires. A partir de la liste trouvée, l'utilisateur peut télécharger les graphes qui l'intéressent.

5.2.1.3 Quelques perspectives

L'application d'une grammaire locale à un texte implique la consultation des dictionnaires lorsque ses étiquettes symbolisent un ensemble de mots (<*N*> pour tous les noms). Si le graphe utilise des mots techniques qui ne se trouvent pas dans le dictionnaire du système, alors il pourra être utile d'ajouter le dictionnaire associé à la grammaire lors de l'insertion de cette dernière dans la bibliothèque[112]. Par ailleurs, les grammaires servent aussi à appliquer le contenu de tables syntaxiques : il faudra donc donner la possibilité d'insérer des tables syntaxiques.
Nous pourrions également réaliser des recherches approximatives de séquences à l'aide des outils de l'équipe de F. Guenthner (CIS, Université de Munich). Par ailleurs, on pourrait imaginer qu'un utilisateur ayant entamé une description sous la forme d'un graphe veuille savoir s'il n'existe pas dans la bibliothèque un graphe similaire (cf. section sur l'intersection de grammaires).

5.2.2 Fonctionnement général

Notre système a une architecture distribuée client-serveur (cf. Gardarin, 1999) permettant d'accumuler dans un même lieu des grammaires locales, construites à divers endroits géographiques. Ce système permet par ailleurs aux chercheurs du domaine de récupérer les informations et données qui les intéressent. Il contient une base de données (des composants linguistiques), un moteur et un système de requêtes permettant de mettre à jour la base de données ou d'accéder aux informations contenues par celle-ci. Dans son aspect général, notre système ne diffère pas des autres systèmes d'information.
Le fonctionnement général de ce système est également très classique (cf. schéma ci-dessous). Un utilisateur situé du côté client montre son désir d'envoyer une requête via l'interface. Les données nécessaires au traitement de cette requête sont alors réunies et le client construit la requête désirée au bon format à l'aide d'un module de construction de requêtes. Puis, le client se charge d'envoyer cette requête au serveur via Internet. Là, un module de réception de requêtes gère le flux des requêtes entrantes en n'en laissant passer qu'une seule à la fois (les autres sont mises en attente). Le gestionnaire des requêtes pré-analyse la requête et la distribue à la procédure adéquate du moteur qui modifie ou extrait des informations de la base de données. Le résultat du traitement est ensuite envoyé au constructeur de réponses (sur le serveur). Celui-ci élabore une réponse formatée qui est envoyée via Internet au client. Ce dernier réceptionne la réponse et l'envoie au gestionnaire des réponses qui, en fonction de la requête, représente d'une certaine manière le résultat sur l'interface.

[112] Remarque de C. Fairon aux Journées Intex 2002.

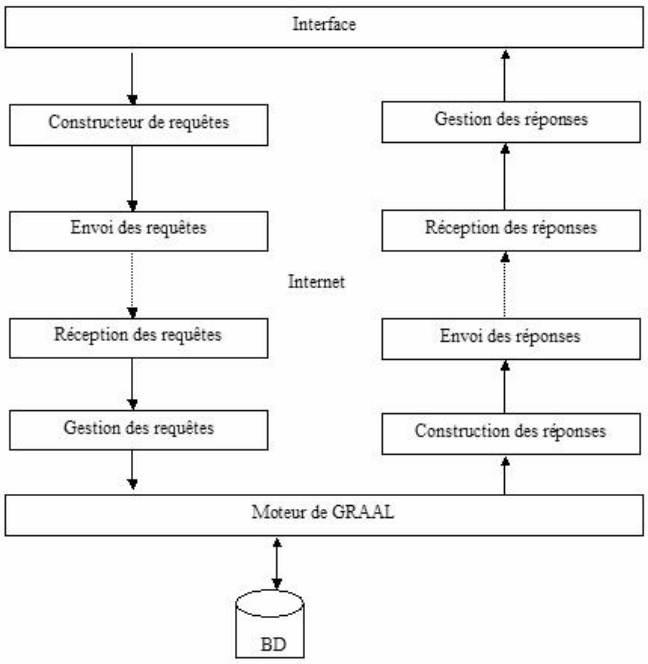

Figure 92 : fonctionnement général de GraAL

Illustrons ce fonctionnement général au moyen d'un exemple. Lors de la session d'ouverture d'une connexion, l'utilisateur saisit son nom *user* et son mot de passe *password* dans deux champs de l'interface. Ces deux séquences sont capturées par le constructeur de requêtes qui confectionne le flux suivant :

CONN *user password*

Ce flux est formé de trois chaînes de caractères séparées de tabulations. La première chaîne indique le nom de la requête, la deuxième correspond au nom d'utilisateur et la troisième désigne le mot de passe. Chaque type de requête a un format particulier. Ce flux est ensuite envoyé au serveur. Le serveur analyse la requête et la traite, tout en interdisant le traitement simultané d'autres requêtes. Il renvoie alors au client le résultat de la requête (connexion acceptée ou non) plus des données générales concernant la bibliothèque (langues, utilisateurs, arborescence générale, etc.) dans un flux de prototype suivant :

CONN <résultat> <rapport> <données générales>
<résultat> : 0 (connexion non acceptée) ou 1 (connexion acceptée)
<rapport> : une chaîne de caractères indiquant les erreurs ou avertissements
<données générales> : un objet comprenant les données générales de la bibliothèque

Après réception et analyse, le client affiche le résultat sur l'interface et si la connexion est acceptée, rafraîchit différents champs de l'interface. Chaque type de requête a un protocole spécifique compris à la fois par le serveur et le client.
Notre système est implanté en java et utilise le codage Unicode pour les caractères. Nous utilisons également certains modules d'Unitex (S. Paumier, 2003).

5.2.3 Base de données

Notre base de données relationnelle[113] représentée dans la figure ci-dessous contient cinq entités représentées par des ellipses :

- les dictionnaires
- les grammaires
- les tables
- les langues
- les utilisateurs

L'entité centrale est celle des grammaires qui contient un ensemble de graphes. Chaque graphe possède plusieurs champs : un nom pointant vers un fichier le représentant explicitement [extension « .grf »] et vers un fichier le documentant [extension « .gdoc »] plus des caractéristiques (si c'est un graphe à sorties, à variables, etc.). L'entité « utilisateurs » comprend l'ensemble des utilisateurs inscrits : les champs concernent des informations personnelles simples (nom d'utilisateur, mot de passe, nom, prénom, courriel, etc.). Les entités sont liées entre elles aux moyens de relations représentées par des rectangles :

- la relation *LangueDeDico* indique la langue associée à chaque dictionnaire ;
- la relation *UtilEstAuteurDeDico* indique les auteurs des dictionnaires ;
- la relation *UtilEstAuteurDeGram* indique les auteurs des graphes ;
- la relation *UtilEstAuteurDeTable* indique les auteurs des tables ;
- la relation *TableUtiliseGram* indique les tables utilisées par les grammaires ;
- la relation *DicoUtiliseGram* indique les dictionnaires utilisés par les grammaires ;
- la relation *UtilEstInterresseParGram* indique les utilisateurs intéressés par les grammaires et qui souhaitent recevoir un courrier électronique quand elles sont modifiées.

Notons que la relation entre une grammaire et une langue est réalisée par transitivité car il existe toujours un dictionnaire associé à une grammaire. Les dictionnaires par défaut sont les dictionnaires du système (dictionnaires DELA).

[113] Pour plus de détails sur les bases de données, se référer à G. Gardarin (1999).

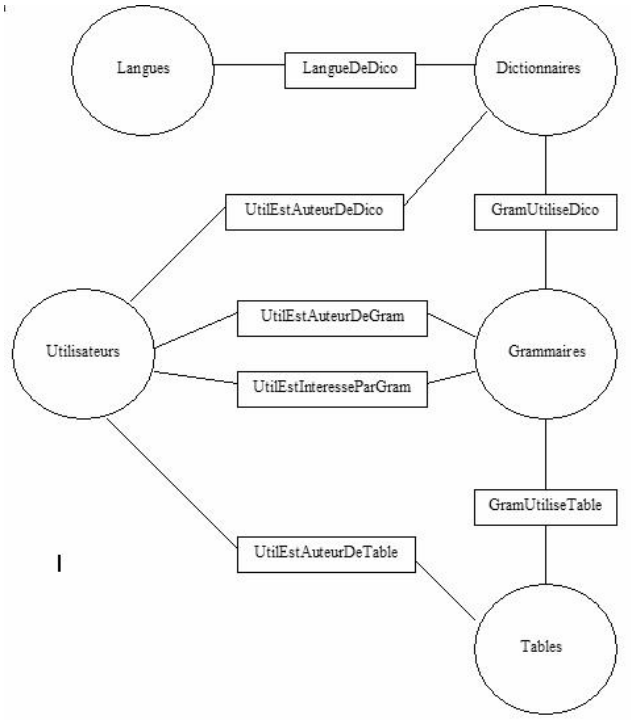

Figure 93 : base de données

5.3 Normalisation des grammaires locales

5.3.1 Représentation théorique des grammaires

Les grammaires locales sont théoriquement équivalentes à des réseaux récursifs de transitions (RTN)[114] [W.A. Woods, 1970]. Les grammaires peuvent donc être définies comme des 4-uplets $<N, T, aut, S>$ où N est un alphabet[115] de symboles non-terminaux, T un alphabet de symboles terminaux, *aut* un ensemble de règle-automates sur $N \cup T$, S l'axiome de départ (ou l'automate principal). Pour tout $X \in N$, il existe, dans l'ensemble des règles-automates *aut*, un automate[116] $aut(X) = <Q(X), q_0(X), F(X), \delta(X), N \cup T>$ où $Q(X)$ est un ensemble d'états, $q_0(X)$ l'ensemble des états initiaux, $F(X)$ l'ensemble des états finaux, $\delta(X) \subseteq Q(X) \times (N \cup T)^* \times Q(X)$ l'ensemble des transitions. Soit la grammaire $G = <N, T, aut, S>$. Etant donné G, chaque symbole non terminal X de N symbolise un sous-automate de G mais aussi une grammaire incluse dans G dont X est l'axiome de départ. Par facilité d'écriture, nous appelons X, l'automate $aut(X)$ dans G. Pour tout automate X de G, si le symbole non terminal Y appartient à son alphabet, c'est-à-dire si l'automate X contient le symbole non-terminal Y, alors l'automate Y est un sous-automate de X. Nous parlerons aussi de sous-automate direct. Notons qu'un symbole non terminal X de G peut être considéré comme inutile lorsque $aut(X)$ n'est pas directement ou indirectement appelé à partir de S.

Soit une collection de n grammaires locales (G_i) avec $G_i = <N_i, T_i, aut_i, S_i>$. Pour tout $X \in N_i$, $aut_i(X) = <Q_i(X), q_{i0}(X), F_i(X), \delta_i(X), N_i \cup T_i>$. Notre bibliothèque de grammaires locales (que

[114] Les graphes à variables (ou de réécriture) ne sont pas des RTN. Nous verrons que lors de la normalisation de ce type de grammaires locales, nous supprimerons les sorties et nous nous ramenons à un RTN.
[115] Un alphabet est un ensemble de symboles élémentaires (ou atomiques).
[116] Pour plus de détails sur les automates, le lecteur est invité à se référer aux ouvrages suivants : M. Gross et A. Lentin (1969) ; T. Sudkamp (1997).

l'on note *B*) est l'union d'un ensemble de grammaires G_1, …, G_n. Ainsi, comme les langages algébriques sont fermés par union, *B* est aussi un RTN. On peut considérer *B* comme le quadruplet $<N_B,T_B, aut_B, S_B>$ tel que :

- N_B est l'union des alphabets de non terminaux N_i auxquels on adjoint un nouveau symbole S_B, l'axiome de *B*;
- T_B est l'union des alphabets de terminaux T_i ;
- Aut_B est l'union des ensemble Aut_i, plus la règle-automate associée à l'axiome S_B.

La règle-automate $aut_B(S_B)$ reconnaît les axiomes S_i avec $i \in [1,n]$, ainsi, on a :

$$L(aut_B(S_B)) = \{S_i \mid i \in [1,n]\}$$

$L(aut_B(S_B))$ est le langage reconnu par $aut_B(S_B)$. Nous appellerons $aut_B(S_B)$ l'automate d'union. Nous gardons le terme d'automates principaux de *B* pour $S_1, S_2, \ldots, S_n$.

Exemple : Soit une bibliothèque *B* qui comprend deux grammaires locales $G_1 = <\{X,Y\},\{a,b\},aut_1,S_1>$ et $G_2 = <\{Y\},\{a,b,c\},aut_2,S_2>$ avec aut_1 et aut_2 représentés dans les figures suivantes[117] :

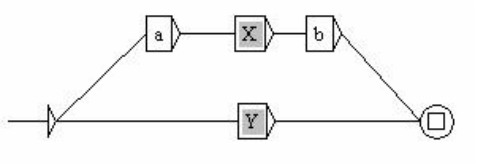

Figure 94 : S1

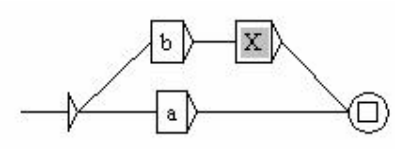

Figure 95 : X

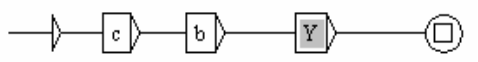

Figure 96 : S2

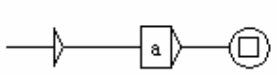

Figure 97 : Y

La séquence *aXb* appartient au langage $L(aut(S_1))$. La séquence *abab* est reconnue par la grammaire S_1 et donc elle appartient à $L(G_1)$, langage sur $\{a,b\}$ reconnu par G_1.

Ainsi, on a $N_B = \{S_B, S_1, S_2, X, Y\}$, $T_B = \{a, b, c\}$ et l'automate $aut(S_B)$ est représenté par le graphe suivant :

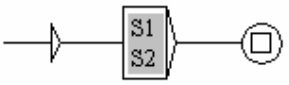

Figure 98 : S_B

La définition de *B* ci-dessus n'est valable que si les bijections associant les automates aux non-terminaux s'étendent en une même bijection : pour tout $X \in N_i$ et pour tout $Y \in N_j$, si $X = Y$ alors $aut_i(X) = aut_j(Y)$. De manière informelle, cela revient à dire que si deux symboles non terminaux de deux grammaires différentes sont les mêmes, alors les règles-automates

[117] Nous représentons les automates d'une manière un peu spéciale (M. Silberztein, 1993). Les graphes se lisent de gauche à droite. Les transitions étiquetées se trouvent dans les boîtes. Les étiquettes non terminales sont grisées et représentent des appels à des sous-graphes de même nom. Les états ne sont pas représentés sauf l'état initial et l'état final.

associées sont les mêmes. Nous verrons plus tard que, lors de la procédure d'insertion de grammaires dans la bibliothèque, ce n'est plus forcément le cas.

Remarque importante :

Jusqu'à ce chapitre, nous avions utilisé le mot « graphe » pour ce que nous appelons maintenant automate. Ces deux notions sont habituellement utilisées de façon équivalente par les utilisateurs d'Intex et d'Unitex mais, dorénavant, nous ne parlerons plus que d'automates pour ce qui concerne les règles des grammaires locales car, un peu plus tard, nous utiliserons le terme de graphe pour dénoter d'autres types de données.

5.3.2 Normalisation des grammaires en pratique

Les grammaires locales susceptibles d'être stockées dans la bibliothèque sont construites à l'aide des éditeurs de graphes d'Intex et d'Unitex. Les deux formats sont très proches mais leur système de codage des caractères est différent : Unitex utilise le codage Unicode 3.0 standard UTF-16 Little Indian (1 caractère = 2 octets) et Intex utilise un codage ASCII (1 caractère = 1 octet). On peut aussi imaginer que l'on veuille utiliser des grammaires d'un autre type telles que les grammaires algébriques (dans un format de fichier particulier).
Afin d'avoir des procédures de traitement les plus efficaces possibles, nous décidons de représenter les grammaires de manière standard dans notre système. Nous gardons une copie de chaque grammaire dans sa version d'origine pour que cette dernière puisse être utilisée dans le logiciel pour lequel elle a été construite. Cela étant dit, il est nécessaire de normaliser les grammaires en se ramenant à une représentation la plus proche possible de la théorie et en supprimant les informations inutiles. Les formats choisis sont les plus puissants. Ainsi, nous décidons de coder les caractères en Unicode afin de pouvoir traiter toutes les langues. Le processus de conversion d'une grammaire ASCII en une grammaire Unicode ne pose pas de problème majeur car il existe des procédures établies. Les grammaires sont représentées sous la forme de RTN[118]. Dans le cas où l'on veuille utiliser des grammaires algébriques, il conviendra de convertir ces dernières dans le bon formalisme : par exemple, au moyen de l'algorithme de Nederhof (2000).
La normalisation des grammaires dépend entièrement de ce que nous voulons en faire. Notre système doit contenir des outils de recherche d'information dans ces grammaires et plus exactement sur leur contenu linguistique. La représentation graphique n'a aucune utilité pour ces outils. Elle peut donc être supprimée. Par ailleurs, les grammaires que nous traitons sont des grammaires d'analyse linguistique dont les automates n'ont pas de sortie. Ainsi, nous décidons d'enlever toutes les informations de sortie des grammaires qui en contiendraient. L'information de l'existence de telles sorties est extraite automatiquement à partir de l'analyse automatique des grammaires à insérer et placées dans des champs spéciaux de la table contenant les informations sur les grammaires.
Les éditeurs d'automates offrent une grande liberté aux auteurs des automates. Par exemple, les auteurs peuvent construire des transitions dont l'étiquette comprend plusieurs symboles consécutivement. Pour faciliter les traitements informatiques, chaque transition comprenant une séquence de *n* symboles simples est découpée en *n* transitions d'éléments simples. Il existe de multiples opérations de normalisation des transitions. Certaines étiquettes apportent peu d'informations par rapport à la complexité qu'elles génèrent dans les opérations. Elles sont alors modifiées. Par exemple, l'étiquette # qui interdit l'espace entre deux symboles est

[118] Théoriquement, il existe des représentations formelles plus puissantes pour décrire le langage naturel. Mais nous nous plaçons dans le cadre théorique du lexique-grammaire qui représente les faits linguistiques de manière très simple.

remplacée par l'étiquette vide[119]. On perd dans ce cas des informations mais elles ne sont pas indispensables.
Des procédures classiques de synchronisation des automates permettent de supprimer les transitions étiquetées par le mot vide des règles-automates. L'émondation supprime les états inutiles et la minimisation rend notre représentation optimale (cf. T. Sudkamp, 1997).
Pour se rapprocher de la représentation théorique, il convient d'extraire des automates les deux alphabets. La distinction entre les symboles non terminaux et les symboles terminaux est immédiate : le symbole « : » sert de préfixe à chaque symbole non terminal (c'est-à-dire une chaîne de caractère qui désigne le nom d'un sous-automate).
Un point cependant pose problème. L'alphabet des éléments terminaux ne comporte pas que des symboles élémentaires. En effet, si l'on se place au niveau linguistique, on constate que la grande majorité des étiquettes employées ne sont pas atomiques, c'est-à-dire qu'elles représentent un ensemble de symboles élémentaires. Nous examinons chaque type d'étiquettes au cas par cas.

- les chiffres et les caractères non-alphabétiques

Ces symboles sont par définition minimaux, donc ils ne posent pas de problèmes. Exemples : « 1 », « 2 », …, « + », « - ».

- les séquences de lettres :

Une séquence de lettres délimitée par deux séparateurs forme un mot graphique. D'un point de vue non linguistique, une telle séquence peut former une unité atomique. Cependant, d'un point de vue linguistique, un mot sous sa forme graphique peut être considéré comme un ensemble de mots (ou d'étiquettes lexicales) ayant la même forme graphique (fléchie) dans les dictionnaires. Par exemple, si l'on prend le mot *donne*, il y a deux analyses possibles : soit c'est un nom qui a pour forme canonique *donne*, soit c'est un verbe qui a pour forme canonique *donner*. Le nombre d'éléments (ou cardinal) de cet ensemble dépend de la finesse de codage du dictionnaire. Dans un dictionnaire où chaque entrée n'est associée qu'à une catégorie grammaticale, le cardinal du mot *danse* [noté card(*danse*)] serait égal à 2. Dans un dictionnaire contenant aussi des informations flexionnelles comme le DELAF, card(*donne*) est plus élevé et est égal à 7 comme le montre le petit échantillon de dictionnaire ci-dessous :

donne,.N+z1:fs
donne,donna.N+z3:fp[120]
donne,donner.V+z1:P1s:P3s:S1s:S3s:Y2s

Dans cet échantillon, chaque ligne désigne une entrée lexicale. La première séquence de caractères délimitée à droite par une virgule est la forme fléchie. La deuxième délimitée à gauche par la virgule et à droite par un point est la forme canonique. Quand cette séquence est vide, cela signifie que la forme canonique est la même que la forme fléchie. Le code juste après le point est le code grammatical associé à l'entrée lexicale (ex : *N* pour nom, *V* pour verbe). Les séquences de lettres et de chiffres après un « + » sont des traits de diverses valeurs (ici, *+z1* indique que l'entrée lexicale fait partie du langage courant). Les séquences de lettres et de chiffres après un « *:* » sont des informations flexionnelles (par exemple, *:P1s* signifie que l'entrée est à la première personne du singulier du présent de l'indicatif). Notons

[119] Un travail différent mais proche a été réalisé dans M.Constant (2001) pour construire le compilateur du logiciel AGLAE (S. Paumier, 2000).
[120] *+z3* est un trait indiquant que ce mot fait partie d'un discours très spécialisé.

que plusieurs entrées lexicales peuvent être codées sur une seule ligne si le seul champ qui les distingue est le code flexionnel comme c'est le cas pour la dernière ligne de l'exemple ci-dessus. Dorénavant, nous utilisons comme dictionnaires de référence les dictionnaires de type DELAF.

- les unités linguistiques non ambiguës

Dans Intex et Unitex, il est possible de définir des étiquettes non ambiguës. Ce sont des séquences entre accolades qui définissent avec précision une entrée lexicale comme dans un dictionnaire. Par exemple, l'étiquette *{danse,danse.N:fs}* indique clairement que l'on a la forme graphique *danse* qui provient de la forme canonique *danse* et qui est un nom au féminin singulier[121].

- les étiquettes ensemblistes (lexico-syntaxiques)

Certaines étiquettes sont des abréviations d'ensembles d'unités linguistiques élémentaires. Elles sont écrites dans Intex et Unitex entre angles. On distingue parmi celles-ci, les étiquettes ensemblistes lexicales et grammaticales. Les étiquettes lexicales doivent obligatoirement contenir une forme canonique. Par exemple,

<*tendre*> désigne tous les mots dont la forme canonique est *tendre*
<*bleu.N*> désigne tous les noms (*N*) dont la forme canonique est *bleu*

Les étiquettes grammaticales désignent des ensembles à partir de codes grammaticaux, de codes flexionnels et de traits syntaxiques et sémantiques. Par exemple,

<*A:fp*> désigne tous les adjectifs (*A*) au féminin pluriel (*fp*)
<*N+Hum*>[122] désigne tous les noms qui ont le trait sémantique humain *+Hum*

- les méta-étiquettes :

Les métas désignent également des ensembles de mots. Les caractéristiques sont plutôt graphiques : <*MOT*> désigne tous les mots (chaîne de caractères entre deux séparateurs) ; <*PRE*> désigne tous les mots commençant par une majuscule. A noter, cependant, qu'il existe une méta-étiquette qui utilise des informations linguistiques : <*DIC*> qui désigne tous les mots qui se trouvent dans le (ou les) dictionnaire(s) courant(s).

L'étude de ces différents cas montre clairement que les symboles terminaux utilisées dans les grammaires ne sont pas élémentaires. Théoriquement, il est toujours possible de se ramener à des symboles atomiques en remplaçant explicitement chaque symbole ensembliste par l'ensemble des unités linguistiques qu'il désigne. Les symboles ensemblistes lexicaux (qui contiennent un élément lexical dans leur définition : par exemple, <*maison*>) ne représentent qu'un ensemble restreints d'unités lexicales (au maximum, une quarantaine d'éléments [les verbes]). La procédure proposée consiste juste à supprimer la transition contenant ce symbole et à la remplacer par autant de transitions qu'il y a d'éléments dans l'ensemble. Les symboles ensemblistes grammaticaux posent des problèmes d'ordre pratique. En effet, prenons le symbole <*N*> qui désigne tous les noms. L'application de la précédente procédure

[121] Le format est le même que celui d'une entrée lexicale au format DELA (B. Courtois et al., 1990).
[122] Dans Unitex, l'étiquette <N-Hum> qui désigne tous les noms qui n'ont pas le trait humain est autorisée. Notre système ne traite pas ce type d'étiquettes pour l'instant.

consisterait à remplacer cette étiquette par l'ensemble de tous les noms. Le cardinal de cet ensemble étant d'une centaine de milliers, la normalisation proposée est clairement coûteuse en espace mémoire. Le phénomène est amplifié avec les méta-étiquettes. En effet, l'ensemble des mots est gigantesque à cause de la production (infinie) de nouveaux mots comme les noms propres. La procédure est alors impossible à mettre en œuvre[123]. Nous décidons donc de laisser en l'état les symboles ensemblistes non lexicaux. Ainsi, les symboles terminaux de nos grammaires normalisées ne sont pas tous élémentaires, ce qui pose des problèmes pour la comparaison entre les étiquettes des automates.

5.3.3 Les symboles terminaux

Le problème est maintenant de représenter et normaliser les symboles terminaux de manière à rendre efficace la procédure de mise en correspondance entre deux étiquettes terminales (tag matching).

5.3.3.1 Normalisation des terminaux

D'abord, un symbole terminal *A* peut être représenté au moyen d'un découpage de l'information en *n* champs indépendants A_1, A_2, …, A_n.[124] Si l'on prend les dictionnaires de type DELA comme référence, nous pouvons découper chaque étiquette en cinq champs indépendants :

- un champ correspondant à la forme graphique
- un champ correspondant à la forme canonique
- un champ correspondant au code grammatical
- un champ correspondant aux traits autres que flexionnels
- un champ correspondant aux informations flexionnelles

Notons qu'il faudrait aussi rajouter un champ pour les métas qui requièrent des procédures particulières. Ce champ pourrait simplement indiquer la méta-étiquette utilisée. Cependant, les métas sont clairement dépendantes d'autres champs : par exemple, l'étiquette <*MOT*> implique que la forme graphique soit une suite de caractères. Nous décidons de traiter les métas séparément.
Un symbole terminal *A* peut ainsi être considéré comme l'intersection de ses différents champs (J. Senellart, 1999b) : $A = A_1 \cap A_2 \cap A_3 \cap \ldots \cap A_n$ (dans notre cas, $n \leq 5$). Par exemple, un nom au féminin singulier est bien l'intersection de l'ensemble des noms et de l'ensemble des mots au féminin singulier.
Nous procédons à la normalisation des étiquettes terminales au moyen des dictionnaires de type DELA et des outils de consultation du logiciel Unitex (S. Paumier, 2003). Nous examinons maintenant chaque cas[125].

- les caractères qui ne sont pas des lettres :

La normalisation de ce type d'étiquettes est immédiate comme le montre le tableau ci-dessous qui représente l'étiquette normalisée du chiffre *1*. Celle-ci est découpée en cinq champs :

[123] Sauf si l'on décide de mélanger le niveau morphologique et lexical. Dans ce cas, <MOT> peut être représenté par l'expression régulière <L><L>* représentant une séquence de lettres quelconques d'au moins un élément.
[124] Nous n'avons pas pris la représentation d'E. Roche (1993) qui représente une entrée au moyen d'un automate où chaque code est placé sur une transition. Nous avons préféré utiliser une représentation plus objet mieux adaptée au langage JAVA.
[125] Comme indiqué précédemment, nous ne traitons pas les méta-étiquettes. Leur normalisation consiste uniquement à rajouter une information indiquant le numéro du méta.

chaque ligne correspond à un champ ; la première colonne contient leurs intitulés ; la deuxième les données. Le signe – indique que le champ correspondant est vide.

Forme graphique (FG)	1
Forme canonique (FC)	-
Code grammatical (CG)	-
Traits (Tr)	-
Flexions (Fl)	-

Table 25 : étiquette normalisée de « 1 »

- les suites de lettres (les mots graphiques)

Un mot graphique représente un ensemble relativement limité d'unités linguistiques (ou entrées lexicales). La consultation des dictionnaires donne explicitement cet ensemble. Une procédure simple d'analyse du résultat de la consultation permet de construire les étiquettes. Par exemple, le mot graphique *donne* est équivalent à l'ensemble des trois étiquettes ci-dessous (cf. l'échantillon de dictionnaire dans la section précédente) :

FG	*donne*
FC	*donne*
CG	*N*
Tr	*(z1)*
Fl	*(f,s)*

Donne
Donna
N
(z3)
(f,p)

donne
donner
V
(z1)
((1,P,s),(3,P,s), (1,S,s),(3,S,s),(2,Y,s))

Table 26 : normalisation de l'étiquette graphique *donne*

Le remplissage des champs *FG*, *FC* et *CG* est immédiat car ce sont de simples chaînes de caractères. Le champ *Tr* est l'ensemble trié des traits. Le champ *Fl* est plus compliqué car c'est un ensemble trié d'ensembles triés de flexions.
Il arrive parfois qu'un mot ne soit pas reconnu par le dictionnaire : c'est un mot inconnu. Un mot inconnu est alors codé comme un mot sans catégorie grammaticale, sans information flexionnelle ou sémantique, juste avec une forme graphique.

- les unités élémentaires non ambiguës

La normalisation des étiquettes de ce type est aussi immédiate.

- les symboles ensemblistes lexicaux (ex : <*manger*>, <*bleu.N:f*>)

Un symbole ensembliste lexical doit contenir obligatoirement la forme canonique (*X*) de laquelle dérivent toutes les unités élémentaires contenues dans l'ensemble associé. Ce symbole peut optionnellement comporter un code grammatical, des traits et des informations flexionnelles. Ces informations optionnelles servent à filtrer l'ensemble des unités linguistiques qui ont pour forme canonique *X*. La première étape du processus de normalisation consiste à consulter le dictionnaire de type DELA. Cette consultation fournit la liste des entrées du dictionnaire ayant *X* pour forme canonique. Ensuite, on réalise le filtrage[126] de cette liste à partir des informations optionnelles, puis on construit les étiquettes normalisées à partir du résultat du filtrage. Par exemple, la normalisation de l'étiquette <*bleu.N:f*> se déroule comme suit :

[126] Le fonctionnement du filtrage n'est pas détaillé ici ; il est expliqué un peu plus loin avec le « tag matching ».

⇨ consultation du dictionnaire pour la forme canonique *bleu* :

bleu,.A+z1:ms
bleu,bleue.A+z1:fs[127]
bleu,bleus.A+z1:mp
bleu,bleues.A+z1:fp
bleu,.N+z1:ms
bleu,bleue.N+z1:fs
bleu,bleus.N+z1:mp
bleu,bleues.N+z1:fp

⇨ filtrage de la liste précédente suivant les informations optionnelles (*.N:f*) :

bleu,bleue.N+z1:fs
bleu,bleues.N+z1:fp

⇨ construction des étiquettes normalisées correspondantes à chaque ligne de la liste ci-dessus :

bleues
bleu
N
(z1)
((f,p))

bleue
bleu
N
(z1)
((f,s))

Table 27 : normalisation de l'étiquette <*bleu.N:f*>

- les symboles ensemblistes grammaticaux (ex : <*N+z1+Hum:ms*> ; <*V:P1s*>)

Nous construisons une seule étiquette normalisée par étiquette ensembliste non lexicale. Il suffit d'extraire des étiquettes d'origines les informations à intégrer dans les différents champs du symbole terminal normalisé. La normalisation de <*N+z1+Hum:ms*> donne le résultat suivant :

-
-
N
(Hum,z1)
((m,s))

Table 28 : normalisation de <*N+z1+Hum:ms*>

5.3.3.2 Mise en correspondance d'étiquettes

Théoriquement, la mise en correspondance (« Tag matching ») de deux symboles terminaux est directe car ce sont des symboles élémentaires. La procédure peut se résumer par les quelques lignes triviales ci-dessous :

[127] L'emplacement des formes canoniques et des formes fléchies est permuté dans ce cas-là (cf. S. Paumier, 2002).

```
Fonction match(a,b)
    si a = b alors résultat ← vrai
    sinon résultat ← faux
    finSi
finFonction
```

En pratique, dans nos grammaires, la mise en correspondance de deux symboles terminaux normalisées n'est pas directe car les symboles ne sont pas forcément atomiques comme on l'a vu dans la section précédente. Dans cette partie, nous implémentons une procédure de mise en correspondance de deux symboles terminaux normalisés. Cette procédure est symétrique en *a* et *b*.
Nous définissons des procédures de mise en correspondance pour chaque type de champ. Il en existe trois : un pour *FG*, *FC* et *CG* ; un autre pour *Tr* ; un autre pour *Fl*. Nous implantons donc trois procédures : *matchChampSimple*(*x*,*y*) pour *FG*, *FC* et *CG* ; *matchEnsemble*(*x*,*y*,*grammatical*,*Autom*) pour *Tr* et *matchFlexions*(*x*,*y*,*grammatical*,*Autom*) pour *Fl*. Chacune de ces procédures prend en arguments deux champs (plus d'autres paramètres) et renvoie un booléen. Nous définissons nos procédures de telles manière que, si l'une des procédures de mise en correspondance des champs renvoie faux alors la fonction *match*(*A*,*B*) renvoie faux. Elle renvoie vrai dans les autres cas.

La fonction *matchChampSimple*(*x*,*y*) est une simple comparaison des chaînes de caractères *x* et *y*. Si les chaînes correspondent le résultat est vrai sinon il est faux. Le seul point délicat est lorsque l'un des champs ou les deux sont vides (ou nuls) [x ou y = -]. Dans ce cas, le résultat est vrai. On notera qu'avec cet algorithme, *1* (CG vide) est mis en correspondance avec <*N*> (FG vide). Pour éviter cela, il suffira de vérifier au préalable si l'on n'a pas ce type de cas particuliers.

La fonction *matchEnsemble*(*x*,*y*,*grammatical*,*Autom*) sert à comparer deux champs correspondant aux traits et sera également utilisée comme sous-procédure de la comparaison de l'ensemble des codes de flexions. Si au moins l'un des deux arguments *x* ou *y* est vide alors on retourne vrai. Les symboles *x* et *y* représentent des ensembles triés respectivement (x_1,x_2,…,x_n) et (y_1,y_2,…,y_m). Si l'un des deux est inclus dans l'autre, on retourne également vrai : par exemple, si on a un champ (*Hum*,*z1*) et un champ (*z1*) alors on retourne vrai.
Si les conditions précédentes ne sont pas remplies, les deux champs correspondent si l'union des deux ensembles forme une suite d'informations qui se trouve dans le dictionnaire. Par exemple, si l'on a une étiquette <*N*+*Top*> et une étiquette <*N*+*NPR*>[128] alors ces deux étiquettes correspondent car les deux traits peuvent se trouver dans une même entrée du dictionnaire :

Tours,.N+NPR+Top+Ville

Il convient donc de pré-calculer l'ensemble des séquences possibles de traits se trouvant dans les dictionnaires (en tenant compte que chaque séquence est triée). Cet ensemble est représenté sous la forme d'un automate minimal *Autom*. La construction de cet automate est très simple : pour chaque entrée lexicale du dictionnaire, on construit l'automate reconnaissant l'ensemble des séquences de traits possibles puis on réalise l'union de ces automates et on minimise. Par exemple, si l'on prend l'entrée lexicale ci-dessus (*Tours*), l'automate (non minimal) associé à cette entrée est le suivant :

[128] Les champs *Tr* des deux étiquettes sont respectivement les ensembles triés (*Top*) et (*NPR*). Leur union donne un ensemble trié : (*NPR*,*Top*).

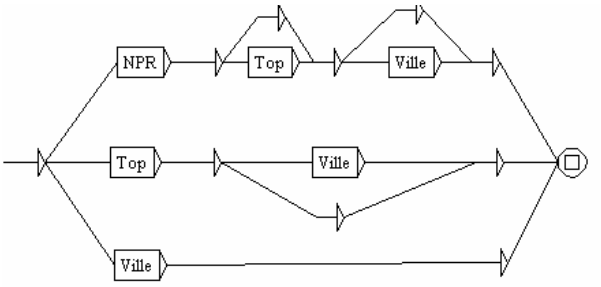

Figure 99 : exemple d'automate pour les traits de l'entrée *Tours*

Il suffira ensuite d'appliquer *Autom* à la séquence d'éléments de l'ensemble formé lors de l'union de *x* et *y*. Par exemple, si l'on reprend nos deux champs précédents, leur union forme l'ensemble trié (*NPR,Top*) soit la séquence *NPR Top* qui est reconnue par l'automate précédent.
Cependant, cette procédure n'est pertinente que lorsque les deux étiquettes contenant *x* et *y* sont ensemblistes et grammaticales (*grammatical* = vrai dans ce cas ; faux autrement). En effet, les étiquettes ensemblistes lexicales normalisées sont équivalentes à des lignes de dictionnaire qui contiennent toutes les informations. Supposons que *u* soit une telle étiquette. Lors de la mise en correspondance avec une étiquette *v*, si un des champs de *v* ne se trouve pas dans *u*, alors *u* et *v* ne correspondent pas. Par exemple, l'étiquette <*Paul*> est normalisée en une ligne de dictionnaire *Paul,.N+NPR:ms*. Si on la met en correspondance avec l'étiquette <*N+Top*>, le résultat de la procédure doit donner un résultat négatif, même si dans le dictionnaire, il est possible d'avoir une autre ligne contenant les traits *+Top* et *+NPR*. Nous résumons cette procédure par l'algorithme ci-dessous :

```
Fonction matchEnsembles(x,y,grammatical,Autom)
    si x ou y est vide alors
        retourner vrai
    finSi
    si x est inclus dans y alors
        retourner vrai
    finSi
    si y est inclus dans x alors
        retourner vrai
    finSi
    si grammatical est vrai et x∪y est reconnu par Autom alors
      retourner vrai
    finSi
    retourner faux
finFonction
```

Le champ comprenant les codes flexionnels est très particulier car il comprend un ensemble trié de codes flexionnels (un code flexionnel étant lui-même un ensemble trié de caractères). Deux champs de type *Fl* correspondent lorsque l'on trouve deux codes flexionnels qui correspondent (cela correspond à un OU logique). Par exemple, les étiquettes <*V:P1s:P2s:I1s*> et <*V:J1p:J2p:I1s*> ont le code flexionnel *I1s* en commun. La mise en correspondance de deux codes flexionnels[129] c_1 (ex : (*p*)) et c_2 (ex : (*m*,*p*)) revient à appliquer la fonction *matchEnsemble*. En effet, l'inclusion de c_1 dans c_2 ou l'inclusion de c_2 dans c_1 indique que les deux codes correspondent. De même que précédemment si l'union des deux

[129] Rappel : le code *:mp:ms* est un ensemble trié de codes flexionnels : ((*m*,*p*),(*m*,*s*)).

codes se trouve dans le dictionnaire, alors il y a correspondance : par exemple, les étiquettes <*N:m*> et <*N:s*> correspondent car le code *ms* se trouve dans le dictionnaire :

homme,.N+z1:ms

Nous résumons cette procédure par l'algorithme ci-dessous :

```
Fonction matchFlexions(x,y,grammatical, Autom)
    pour chaque élément a de x
            pour chaque élément b de y
                    si(matchEnsembles(a,b,grammatical,Autom)) retourner vrai
            finPour
    finPour
    retourner faux
finFonction
```

Nous ne rentrons pas dans les détails pour le cas du « tag matching » mettant aux prises un méta-symbole et un symbole quelconque *x*. Nous donnons juste deux exemples. Supposons d'abord que l'on souhaite comparer <*MOT*> avec un symbole non méta. Il suffit simplement de regarder si le champ FG est une chaîne de caractère. Si ce champ est vide, il y a obligatoirement correspondance. Ensuite, supposons que l'on ait les métas <*DIC*> et <*PRE*>[130], nous supposerons alors qu'il y a correspondance car le dictionnaire comprend des mots commençant par une majuscule.

Il existe un cas que nous n'avons pas traité. Les mots composés pourraient faire l'objet d'une étude plus poussée. En effet, une étiquette <*N*> pourrait très bien correspondre à un nom composé tel que *cordon bleu* de forme de surface Nom Adjectif. Ainsi, dans nos automates une transition étiquetée <*N*> pourrait correspondre à deux transitions consécutives respectivement étiquetées *cordon* et *bleu* (ou même <*N*> et <*A*>). La résolution de tels phénomènes est clairement non triviale et mériterait une étude approfondie. Cependant, nous estimons que son intérêt pour notre application est limité car la mise en correspondance sert simplement à faire des rapprochements entre des séquences d'étiquettes et des grammaires. Les procédures de « tag matching » permettent donc certaines approximations. Lorsque notre système rencontre des étiquettes lexicales se rapportant à un mot composé (ex : <*cordon bleu*>), la procédure de normalisation reconnaît *cordon bleu* comme un lemme, consulte le dictionnaire et renvoie deux étiquettes normalisées équivalentes aux lignes de dictionnaire :

cordon bleu,.N:ms
cordons bleus,cordon bleu.N:mp

Par ailleurs, notons que certaines applications (cf. J. Senellart, 1999b) se servent d'étiquettes équivalentes à des intersections d'ensembles *(<N> & !<:ms>* où *&* est un ET logique et *!* le NON logique). Nous ne traitons pas ces cas.

5.3.4 Quelques mots sur l'indexation

Pour accélérer les processus, il convient d'indexer la bibliothèque par plusieurs types d'informations. Lors des procédures de stockage ou de recherche, nous aurons besoin d'accéder rapidement à divers éléments dans la bibliothèque. Par exemple, il sera utile d'avoir une vue des grammaires sous la forme de graphes décrivant les dépendances entre leurs

[130] <*DIC*> désigne tout mot se trouvant dans le dictionnaire ; <*PRE*> désigne tout mot commençant par une majuscule.

automates : étant donné un automate, le calcul de la liste des sous-automates qu'il utilise est alors très peu coûteux. Les symboles terminaux doivent être associés à l'ensemble des automates dans lesquels ils apparaissent. Ceci permet par exemple de déterminer immédiatement l'ensemble des grammaires qui contiennent un mot donné. Les automates doivent également être indexées selon les mots contenus dans leurs documentations (mots-clés, textes de description) : cela permet par exemple d'accéder immédiatement aux automates dont la documentation comprend certains mots-clés. Nous fournirons plus de détails sur ces index lors des descriptions des différentes procédures où elles sont utiles. Pour la suite, nous considérons que notre bibliothèque est formée d'un ensemble d'index qu'il faut mettre à jour lorsque la bibliothèque est modifiée.

5.4 Stockage des grammaires

5.4.1 Généralités

Les opérations de stockage des grammaires locales dans la bibliothèque peuvent se résumer à deux opérations :

- l'insertion d'une grammaire locale *G* dans la bibliothèque *B* ;
- la suppression d'une grammaire locale *G* (définie par un symbole non-terminal) incluse dans *B*.

Contrairement aux bases de données classiques, ces opérations ne sont pas triviales. L'insertion pose certains problèmes du fait que *G* et *B* peuvent avoir un symbole non terminal en commun mais qui n'ont pas les mêmes règles-automates associées : lorsqu'il y a conflit de noms. Dans ce cas-là, ces automates n'ont pas forcément le même contenu (nouvelle version). La suppression, quant à elle, est une opération complexe du fait que *G* peut être strictement récursive et qu'elle peut contenir un symbole non-terminal lui-même contenu dans une autre grammaire non incluse dans *G*. Ce problème ressemble beaucoup à un problème de libération automatique d'objets dans un programme. Nous aurons besoin de concepts de la théorie des graphes.

Une grammaire locale *G* peut être ajoutée à *B* ou supprimée de *B* de deux manières :

- soit partiellement, précisément on insère (ou supprime) l'automate principal de la grammaire G sans ses sous-automates ;
- soit on insère (ou supprime) son automate principal et récursivement tous ses sous-automates.

Dans cette partie, nous revenons brièvement sur la théorie des graphes et la notion de graphe de dépendance d'une grammaire. Enfin, nous donnons les algorithmes d'insertion de grammaires locales, puis les algorithmes de suppression d'une grammaire locale. On supposera les grammaires normalisées et la bibliothèque indexée.

5.4.2 Préliminaires

5.4.2.1 Rappels théoriques sur les graphes

Nous présentons quelques bases sur la théorie des graphes. Pour plus de détails, le lecteur est invité à lire N. Christofides (1975), A.V. Aho et al. (1983), R. K. Ahuja et al. (1993) ou T. Sudkamp (1997) qui nous ont inspiré même si certaines notations nous sont personnelles. Un graphe orienté *G* est défini par le 2-uplet $< V, E >$ avec *V* un ensemble de sommets[131] et *E* un ensemble de relations entre ces sommets (que l'on appelle aussi arcs[132]). Soit $a \in E$, alors *a* est un couple ordonné (s_1,s_2) avec $s_1 \in V$ et $s_2 \in V$. Désormais, nous abrégeons le terme

[131] *V* pour vertex en anglais.
[132] *E* pour edge en anglais.

graphe orienté en graphe. Graphiquement, un sommet correspond à un petit cercle (ou autre figure géométrique). Un arc (s_1,s_2) est une flèche allant du sommet s_1 au sommet s_2. Illustrons cela par un exemple : soit $G_1 = <V_1,E_1>$ un graphe avec $V_1 = \{1,2,3,4\}$ et $E_1 = \{(1,3),(1,4),(2,1),(4,3)\}$ (cf. la figure ci-dessous). Nous donnons également l'équivalent de ce graphe au format Intex et Unitex: les sommets sont des rectangles étiquetés. Les arcs sont les traits qui relient ces sommets. Leur sens d'orientation est indiqué par la façon dont l'arc est relié aux sommets : sur le sommet de départ, l'arc est attaché à droite.

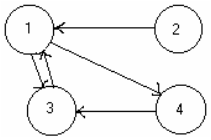

Figure 100 : exemple de graphe

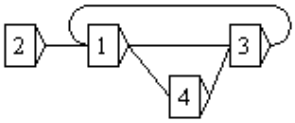

Figure 101 : graphe au format Unitex

Un chemin dans un graphe est une séquence d'arcs $a_1, a_2, \ldots, a_n$ où, pour chaque $i \in [1,n]$, $a_i = (x_j,x_{j+1})$ avec, pour chaque $j \in [1,n]$, $x_j \in V$. On peut simplement dire qu'un chemin est une séquence d'arcs consécutifs. Par exemple, dans le graphe G_1, la séquence d'arcs (2,1) (1,4) (4,3) (3,1) est un chemin allant du sommet 2 au sommet 1. Il n'existe pas de chemin allant du sommet 2 à lui-même.
Un graphe G est cyclique si et seulement si il existe au moins un sommet u de G tel qu'il existe un chemin allant de u à lui même. Par exemple, le graphe G_1 est cyclique car il existe un chemin allant du sommet 1 au sommet 1.
L'ensemble des sommets accessibles à partir d'un sommet x comprend x plus tous les sommets y tels qu'il existe un chemin allant de x à y. Pour la suite, on note cet ensemble $\Gamma(x)$. Par exemple, dans le graphe G_1, $\Gamma(1) = \{1,3,4\}$; $\Gamma(2) = \{1,2,3,4\}$; $\Gamma(3) = \{1,3,4\}$; $\Gamma(4) = \{1,3,4\}$. L'ensemble des sommets co-accessibles d'un sommet x comprend x plus tous les sommets y tels qu'il existe un chemin allant de y à x. Pour la suite, on note cet ensemble $\Phi(x)$. Dans G_1, $\Phi(1) = \{1,2,3,4\}$; $\Phi(2) = \{2\}$; $\Phi(3) = \{1,2,3,4\}$; $\Phi(4) = \{1,2,3,4\}$.

Nous donnons maintenant la définition d'une composante fortement connexe, notion fondamentale pour la suppression d'une grammaire dans la bibliothèque. La composante fortement connexe contenant le sommet x d'un graphe G est définie par l'ensemble :

$$CFC(x) = \{y \in V \mid y \in \Gamma(x) \text{ et } y \in \Phi(x)\}$$

Cet ensemble contient tous les sommets qui sont à la fois accessibles et co-accessibles du sommet x. A partir de la définition précédente, $CFC(x)$ peut clairement s'interpréter comme l'intersection de l'ensemble des sommets accessibles de x avec l'ensemble des sommets co-accessibles de x :

$$CFC(x) = \Gamma(x) \cap \Phi(x)$$

Son calcul est donc très facile à mettre en œuvre. La relation $y \in \Gamma(x) \cap \Phi(x)$ est une relation d'équivalence et elle définit des classes appelées composantes fortement connexes. Il existe par ailleurs des algorithmes qui calculent toutes les composantes fortement connexes d'un graphe en temps linéaire. Le plus connu d'entre eux est l'algorithme de Tarjan (1972) mais il en existe d'autres (M.Sharir, 1981 ; E. Nuutila, 1993, etc.).

On peut associer chaque sommet d'un graphe à une composante fortement connexe. Dans le graphe G_1, par exemple, il existe deux composantes fortement connexes : {1,3,4} et {2}. Si on le souhaite, on peut même construire ce que l'on appelle le graphe condensé du graphe G. Dans ce graphe $G^* = <V^*,E^*>$, chaque sommet de V^* est une composante fortement connexe de G et un arc (x^*,y^*) existe ($\in E^*$) si et seulement si il existe un arc (x,y) dans G pour lequel $x \in x^*$ et $y \in y^*$ (avec $x^* \neq y^*$). Ce nouveau graphe n'est pas cyclique. Par exemple, le graphe condensé de G_1 est le graphe $G_1^* = <V_1^*,E_1^*>$ où $V_1^* = \{x_1,x_2\}$ avec $x_1 = \{1,3,4\}$ et $x_2 = \{2\}$, et $E_1^* = \{(x_2,x_1)\}$ car $2 \in x_2$, $1 \in x_1$ et $(2,1) \in A$. G_1^* est donné dans la figure ci-dessous.

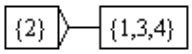

Figure 102 : graphe condensé de G_1

5.4.2.2 La notion de graphe de dépendance

On dit qu'un automate X d'une grammaire $G = <N,T,Aut,S>$ est directement dépendant d'un autre automate Y de cette même grammaire, si et seulement si l'automate Y est un sous-automate directement invoqué par une des transitions de X. Un graphe de dépendance d'une grammaire représente les dépendances directes entre les automates de cette grammaire. Chaque sommet du graphe de dépendance d'une grammaire G correspond à un symbole non terminal X (soit un automate). Chaque sommet X est relié à un autre sommet Y par un arc (sens d'orientation : $X \rightarrow Y$) si et seulement si X est directement dépendant de Y. On dit que X est dépendant de Y s'il existe un chemin partant du sommet X qui va au sommet Y. Si une grammaire est strictement récursive, son graphe de dépendance est cyclique. Par exemple, prenons la grammaire suivante : l'alphabet terminal est $\{a,b\}$, l'alphabet non terminal $\{S,X,Y\}$, l'axiome de départ S et l'ensemble des règles-automates est donné ci-dessous :

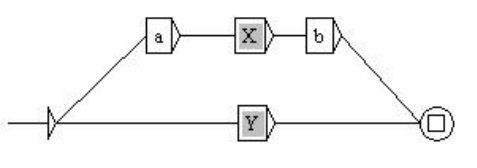

Figure 103 : S

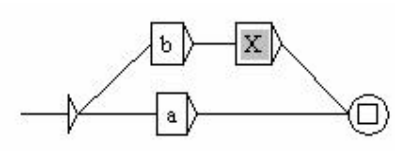

Figure 104 : X

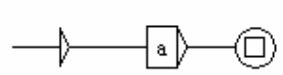

Figure 105 : Y

Le graphe de dépendance correspondant à cette grammaire est le suivant :

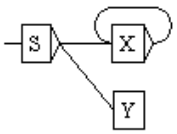

Figure 106 : graphe de dépendance

Dans ce graphe, il y a trois sommets $\{S,X,Y\}$ et trois arcs $\{(S,X),(S,Y),(X,X)\}$. Le couple (S,X) indique qu'il existe un arc entre le sommet S et le sommet X, et donc que *aut(X)* et un sous-automate de *aut(S)*. La grammaire d'exemple est strictement récursive donc le graphe de dépendance est cyclique.

Une application directe de ce graphe de dépendance est, étant donné un automate *X* de la grammaire, de calculer rapidement l'ensemble des sous-automates qu'il utilise (dont il est dépendant). Si l'on code aussi les arcs inverses, il est possible de calculer l'ensemble des automates qui utilisent l'automate *X*. Nous verrons, dans les deux prochaines parties, que l'utilisation des graphes de dépendance est intensive dans notre système.

Remarque importante :
Par la suite, pour simplifier les notations, si *X* est un symbole non terminal d'une grammaire *G*, alors on appellera *X* son sommet associé dans le graphe de dépendance de *G*. Soit *X* un non-terminal d'une grammaire *G*, nous notons *succ(X)* l'ensemble des non-terminaux dont *X* est directement dépendant.

5.4.2.3 Quelques remarques

Les deux entrées fondamentales des procédures de stockage des grammaires sont la bibliothèque *B* et une grammaire *G* (à insérer ou supprimer). La bibliothèque *B* est équivalente à une grammaire locale (union de grammaires) et définie par un 4-uplet $<N_B, T_B, Aut_B, S_B>$. Soit $Gd_B = <V_B, E_B>$ son graphe de dépendance. La grammaire locale à insérer *G* est un 4-uplet <*N*, *T*, *Aut*, *S*>. On suppose que le graphe de dépendance de *G*, noté *Gd* = <*V*, E>, a été pré-calculé.
Les procédures de stockage mettent à jour tous les éléments de la bibliothèque : l'alphabet non-terminal N_B, l'alphabet terminal T_B, l'ensemble des règles automates Aut_B ; les vues partielles de la bibliothèque (les index notés Ind_B et le graphe de dépendance Gd_B). Par ailleurs, dans les algorithmes que nous donnons, nous n'indiquons pas les mises à jour de certaines données telles que « qui est l'auteur ?», « quel est le dictionnaire spécifique associé ? » ou « quel est la table syntaxique associée ? ». Ceci a pour but de faciliter la compréhension du lecteur.

5.4.3 Insertion de grammaires locales

Nous souhaitons insérer une grammaire dans *B*. Nous proposons deux manières de réaliser cette opération : (a) soit on insère uniquement un automate sans ses sous-automates, (b) soit on insère un automate et récursivement tous ses sous-automates. Si la nouvelle grammaire *G* comporte un non-terminal *X* qui existe déjà dans *B*, on souhaite que l'automate *aut(X)* prenne la place de $aut_B(X)$ en tant que nouvelle version du même automate. Dans les deux cas, on suppose que l'automate à insérer est l'automate principal d'une grammaire *G*. En fait, le cas (a) est un cas particulier, à quelques exceptions près, de la méthode (b). L'insertion d'une grammaire ressemble beaucoup à l'union de deux grammaires. La différence principale est au niveau des règles-automates. Si un non-terminal *X* de *G* existe déjà dans *B*, l'union classique conduirait à l'union des deux automates associés à *X* dans *G* et *B*. Si les deux automates sont identiques, on obtient le même automate. Mais s'ils sont différents, l'union diffère de l'insertion car, dans notre système, *aut(X)* est une nouvelle version de $aut_B(X)$. On souhaite seulement que *aut(X)* remplace $aut_B(X)$ dans la bibliothèque. Par ailleurs, le fait que *aut(X)* écrase $aut_B(X)$ a des effets de bord sur l'union des deux alphabets terminaux et sur l'union des alphabets non terminaux. En effet, certains symboles terminaux utilisés dans l'ancienne version $aut_B(X)$ et non utilisés dans le reste de la bibliothèque peuvent avoir disparu dans la nouvelle version *aut(X)*. Il en est de même pour les non terminaux. Ainsi, l'union des alphabets génère du bruit (des symboles non utilisés dans la nouvelle version de la bibliothèque). Nous n'en tenons pas compte car il n'influence en rien les autres procédures. D'ailleurs, les automates associés aux non-terminaux inutiles dans la nouvelle version de *B* pourront toujours intéresser certains utilisateurs du système. Par contre, il est fondamental de tenir compte de ces effets de bord dans les vues partielles de la bibliothèque qui ont besoin

d'informations précises. Ainsi, si un symbole non-terminal *X* existe déjà dans la bibliothèque lors de l'insertion d'une grammaire, il est nécessaire avant de l'insérer de supprimer toutes les informations dans les vues concernant son ancienne version (procédure MiseAJourDesVues, option « suppression »). Après l'insertion effective, il conviendra de mettre à jour les vues partielles (procédure MiseAJourDesVues, option « ajout »). On suppose *G*, *B*, Gd_B, *Gd* et les index *Ind* donnés. Afin de faire une procédure générale pour les deux méthodes (a) et (b), nous ajoutons deux entrées *N'* et *T'*, respectivement l'ensemble des symboles non terminaux associés aux automates à insérer et l'ensemble des symboles terminaux réellement utilisés dans les automates à insérer. Ces informations sont redondantes pour la méthode (b) car *N'* est équivalent à *N* et *T'* est équivalent à *T*. Par contre, pour la méthode (a) comme on n'insère qu'un seul automate de *G*, ces informations sont différentes : *N'* est composé du seul symbole *S* (axiome de départ de *G*) et *T'* est composé de l'ensemble des terminaux utilisés dans *aut(S)*.
Il existe une autre différence entre les deux méthodes. On pose une condition préalable pour le cas (a) alors qu'il n'y en a pas pour (b). Dans (a), les automates desquels l'automate à insérer est directement dépendant doivent déjà exister dans la bibliothèque. C'est-à-dire, si l'un des sous-automates directs de *aut(S)* ne se trouve pas dans *B*, alors on refuse l'insertion. C'est la première condition que l'on doit tester avant toute opération. On notera que cette condition interdit d'insérer *aut(S)* s'il s'appelle lui-même (c'est-à-dire s'il est lui-même un de ses propres sous-automates directs).
La procédure générale marche alors comme suit. On calcule l'ensemble des symboles non terminaux à insérer déjà existants dans la bibliothèque (noté *W*) : on réalise l'intersection entre N_B et *N'*. On met ensuite à jour les vues partielles de *B* en supprimant toutes les données relatives à *W*. Puis, on ajoute l'ensemble *N'* dans l'alphabet des non terminaux de *B* et l'ensemble *T'* dans l'alphabet des terminaux de *B*. On ajoute aussi les automates associés aux éléments de *N'* dans *B* (écrasement s'ils existent déjà). Si l'auteur de *G* souhaite que l'automate principal de *G* soit un automate principal dans *B* (variable booléenne *principal* en entrée de la procédure), on ajoute, dans l'automate d'union de *B* ($aut_B(S_B)$), une transition étiquetée *S* allant de l'état initial à un état final. Enfin, on met à jour les vues partielles de *B* en ajoutant toutes les données relatives à *N'*. Nous synthétisons cette procédure par l'algorithme suivant (procédure principale InsertionGrammaireMain). Dans ces procédures, les variables *G*, *Gd*, *B*, Gd_B et Ind_B sont supposées globales pour alléger le paramétrage des fonctions. La variable d'entrée *option* indique quelle méthode d'insertion est demandée.

```
Procédure InsertionGrammaire(principal,N',T')
    W ← N_B ∩ N'
    MiseAJourDesVues(W, « suppression »)
    N_B ← N_B ∪ N'
    T_B ← T_B ∪ T'
    Pour chaque X ∈ N'
        aut_B(X) ← aut(X)
    finPour
    si principal est vrai alors
        aut_B(S_B).ajouterTransition(q_0(S_B),S,f) avec f ∈ F(S_B)
    finSi
    MiseAJourDesVues(N', « ajout »)
finProcédure
```

```
Procédure InsertionGrammaireMain(principal,option)
    si option = « grammaire entière » alors
        InsertionGrammaire(principal,N,T)
    finSi
    si option = « automate seul » alors
        si un des sous-automates directs de aut(S) n'existe pas dans B alors
            print « ERREUR »
        sinon
            T' ← ensemble des terminaux utilisés dans aut(S)
            InsertionGrammaire(principal,{S},T')
        finSi
    finSi
finProcédure
```

La procédure de mise à jour des vues concerne les différents index Ind_B et le graphe de dépendance de *B*, noté Gd_B. Nous insistons sur la mise à jour du graphe de dépendance. La phase de suppression des données relatives à un ensemble *Ens* de non-terminaux (dans les faits, les non-terminaux de *G* déjà existants dans *B*) est très simple. Il suffit de supprimer pour chaque élément *X* de *Ens*, tous les arcs partant du sommet correspondant. Ainsi, on supprime toutes les dépendances directes des vieilles versions des automates déjà existants. La phase d'ajout des données relatives à un ensemble *Ens* de non-terminaux (*N'*) est un peu plus complexe. Elle consiste d'abord à créer un sommet dans Gd_B pour chaque symbole non terminal dont le sommet correspondant n'existe pas déjà. Enfin, pour chaque sommet correspondant aux symboles dans *Ens*, on fait une copie dans Gd_B des arcs partant du sommet correspondant dans *Gd*. Les algorithmes sont données ci-dessous.

```
Procédure MiseAJourDesVues(Ens,option) //Ens :ensemble de non-terminaux
    IndB.miseAJour(Ens,option)
    miseAJour(Ens,option,GdB,Gd)
FinProcédure


Procédure miseAJour(Ens,option,GdB,Gd)
    si option = « suppression » alors
        pour chaque X ∈ Ens
            GdB.supprimerTousLesArcs(X)  // on supprime tous les arcs partant du sommet X
        finPour
    finSi
    si option = « ajout » alors
        pour chaque X ∈ Ens
            si(X ∉ VB) alors
                GdB.creerSommet(X)
            finSi
        finPour
        pour chaque X ∈ Ens  //pour créer les arcs, il faut d'abord créer les sommets
            pour chaque arc (X,Y) ∈ E
                GdB.creerArc(X,Y)
            finPour
        finPour
    finSi
finProcedure
```

5.4.4 Algorithmes naïfs de suppression de grammaires

5.4.4.1 Préliminaires

Comme pour l'insertion d'une grammaire, il existe deux types de suppressions de grammaires dans B : (a) suppression d'un automate sans ses sous-automates ; (b) suppression d'un automate plus récursivement tous ses sous-automates.
Nous imposons une contrainte préalable pour le cas (a) : il est interdit de supprimer un automate duquel dépend un autre automate. En effet, une telle suppression est « risquée » car elle peut briser des chaînes de dépendance entre automates. Nous donnons cette contrainte afin de garder une certaine cohérence dans la bibliothèque. Soit Z le symbole non terminal associé à l'automate à supprimer. La contrainte est facilement vérifiée en examinant les arcs inverses partant du sommet Z de Gd_B. La suppression de type (a) est très simple. Il suffit de supprimer Z de N_B (et donc $aut_B(Z)$ de Aut_B). Si $aut_B(Z)$ est principal, on supprime, dans $aut_B(S_B)$, la transition étiquetée Z partant de l'état initial. On ne fait rien avec les terminaux car le bruit n'est pas dérangeant. Par contre, la mise à jour des vues partielles est toujours aussi importante. Elle change un peu de celle utilisée précédemment : nous l'appelons MiseAJourDesVuesBis. Elle ne concerne plus que l'option « suppression » (ce qui paraît logique vu la procédure globale dans laquelle nous nous trouvons). La mise à jour des index *Ind* est la même. Mais celle du graphe de dépendance Gd_B est un peu différente. En effet, il faut maintenant supprimer les sommets [133] correspondant à chacun des symboles de l'ensemble des non-terminaux à supprimer *Ens* et pas seulement les arcs partant de ces sommets (comme précédemment).

```
Procédure MiseAJourDesVuesBis(Ens) //Ens :ensemble de non-terminaux
    Ind.miseAJour(Ens, « suppression »)
    GdB.miseAJourBis(Ens)
FinProcédure

//procédure écrite pour n'importe quel graphe de dépendance

Procédure Gd::miseAJourBis(Ens)
    pour chaque X ∈ Ens
        Gd.supprimerSommet(X)
    finPour
finProcédure
```

5.4.4.2 Un algorithme manipulant le graphe de dépendance de la bibliothèque

Examinons le cas (b) qui est nettement plus complexe. Soit G la grammaire à supprimer dans B. G est « définie » par son axiome de départ Z. On dit qu'un automate $aut_B(X)$ est inclus dans G si et seulement si Z est dépendant de X. A contrario, si un automate $aut_B(X)$ n'est pas inclus dans G, on dit qu'il est extérieur à G. Si aucun automate extérieur à G n'est dépendant de l'automate associé à X inclus dans G, X est dit strictement interne à G. Notre objectif est de supprimer tous les automates de G qui sont strictement internes à G. Comme précédemment, notre but est de garder la cohérence des chaînes de dépendance de la bibliothèque.
On peut faire le parallèle avec les problèmes de libération automatique des objets morts[134] dans un programme (cf. Garbage Collector dans la machine virtuelle de Java [JVM]). Les sommets du graphe de dépendance correspondent, dans ce cas-là, aux objets utilisés dans le programme. Les arcs sont des références entre ces objets. Comme les objets peuvent se référencer les uns les autres circulairement, le graphe peut être cyclique. Cependant, les

[133] La suppression d'un sommet *s* implique aussi la suppression des arcs partant de *s* et des arcs atteignant *s*.
[134] Objets morts : objets qui n'ont plus d'utilité dans la suite du programme.

données initiales de ce problème sont différentes du nôtre. En effet, en entrée des procédures de libération d'objets, on connaît les objets qui sont encore concrètement utilisés dans le programme (placés dans des tas ou « heaps »). Les objets encore vivants sont l'ensemble des objets accessibles à partir de l'ensemble des objets dans le tas. Il existe différentes méthodes utilisées par les JVM : par exemple, *stop-and-copy* qui consiste à suspendre temporairement l'exécution du programme et à copier les objets vivants au fur et à mesure que l'on parcourt le graphe des objets vivants ; *mark and sweep* qui consiste à marquer les objets vivants puis à libérer les objets non marqués[135] (cf. B. Eckel, 1998).
Dans notre problème de suppression de grammaires, ce que l'on sait en entrée est que l'on veut libérer l'objet *Z* (axiome de la grammaire *G*) et tous ses sous-automates strictement internes suivant les contraintes énoncées ci-dessus. On peut essayer de se ramener au problème précédent. Pour cela, il faut connaître l'ensemble *H* des automates de *B* desquels aucun automate principal[136] ne soit dépendant (cf. définition d'une grammaire locale) et l'ensemble *I* des automates principaux de *B*. On vérifie préalablement que *Z* est strictement interne à *G*. Soit *J* l'ensemble des automates utilisés par *Z* (*Z* inclus) et qui se trouvent dans $H \cup I$. Puis, à partir de l'ensemble $(H \cup I)$-*J* qui est l'équivalent du tas (ou « heap ») dans la JVM, on marque tous les automates « vivants » de *B* en parcourant *Gd* puis on supprime l'ensemble des automates non marqués. Cette méthode implique que l'ensemble *H* soit mis à jour à chaque insertion de grammaire ou que l'on pré-calcule *H* avant la procédure de suppression. Le temps de calcul de cette procédure est linéaire par rapport à *p* (le nombre de dépendances de *B*). Cette méthode a le grand désavantage de dépendre de la taille de *B*. Plus la bibliothèque grossit et plus cette méthode sera coûteuse. L'idéal serait de travailler sur la grammaire *G* qui a une taille beaucoup plus restreinte que *B*.

5.4.4.3 Un algorithme naïf manipulant le graphe de dépendance de la grammaire à supprimer

Nous proposons maintenant une autre méthode où le raisonnement est inverse : on part des objets que l'on sait morts (ou plutôt qui sont à supprimer) et non des objets vivants comme précédemment. Notre donnée de départ est le symbole non-terminal *Z* qui « définit » une grammaire *G* incluse dans *B*.
Soit Δ un ensemble d'automates inclus dans *G* comprenant $aut_B(Z)$ et un certain nombre d'automates desquels $aut_B(Z)$ est dépendant. Supposons que chaque automate de Δ soit strictement interne à *G*. Ainsi chacun d'entre eux peut être supprimé par application directe de notre contrainte. Soit un automate *A* inclus dans *G* qui n'appartient pas à cet ensemble mais duquel au moins un automate de Δ est directement dépendant. Examinons les différents cas qui s'offrent à nous. Si *A* n'est pas strictement interne à *G*, alors il ne peut être supprimé (contrainte préalable) et il en est de même pour tous les automates desquels il est dépendant. Sinon, on est sûr que *A* peut l'être et donc ajouté à Δ.
On cherche à introduire de nouveaux automates dans Δ jusqu'à être sûr d'avoir trouvé tous les automates inclus dans *G* qui puissent être supprimés.
Illustrons cela par un exemple. Soit une grammaire *G* dont l'automate principal est nommé *Z*. *G* comporte 12 automates nommés *Z*, *X1*, *X2*, …, *X11*. Leurs relations de dépendance sont représentées dans le graphe ci-dessous. Les automates nommés *E1* et *E2* sont extérieurs à *G*. La partie grisée dans la figure de gauche désigne un ensemble Δ d'automates de *G* que l'on peut supprimer. Celle de la figure de droite désigne l'ensemble Δ maximisé. Notre algorithme

[135] Cette méthode est très efficace lorsque le nombre d'objets à libérer est limité.
[136] Par « automate principal dans *B* », on entend chaque automate de *B* dont le non-terminal associé soit contenu dans l'automate $aut_B(S_B)$.

de suppression consiste à partir d'une partie grisée réduite à *Z* et à l'agrandir jusqu'à atteindre une limite.

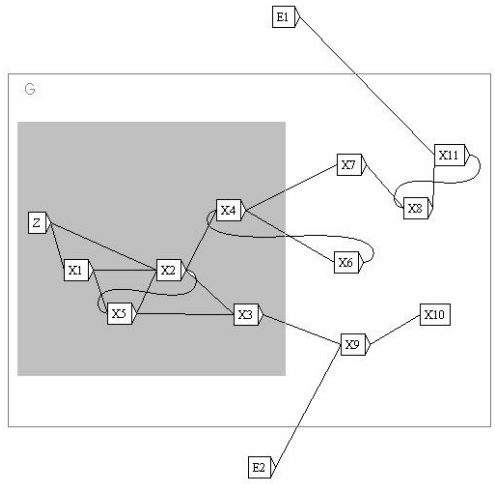

Figure 107 : ensemble Δ en cours de traitement

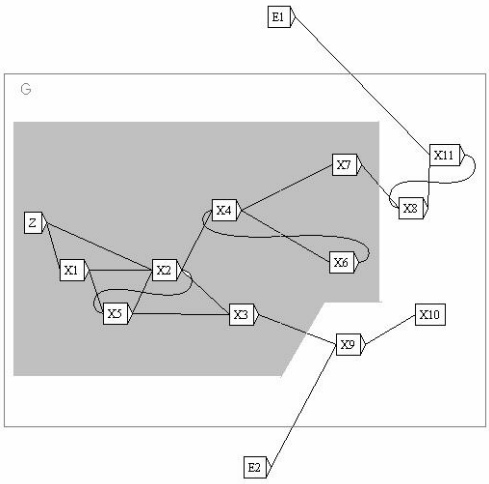

Figure 108 : automates strictement internes à G

Nous commençons par un algorithme naïf qui n'a d'autre intérêt que de suivre précisément l'explication donnée ci-dessus. On vérifie que *Z* est strictement interne à *G*. Le processus débute pour $\Delta = \{Z\}$ et est décrit par l'algorithme ci-dessous (procédure EnsembleASupprimer). On travaille sur le graphe de dépendance de *B*. Les entrées sont donc *Z* et Gd_B. La première étape consiste à calculer les sommets accessibles à partir de *Z*. L'ensemble obtenu est noté *W*. Les sommets à traiter sont mis dans une file. On exécute ensuite les instructions décrites plus haut tant que la file n'est pas vide.

```
Procédure EnsembleASupprimer(Z,GdB)
    Résultat : ensemble Δ
    W ← GdB.accessibilité(Z)
    Δ ←{Z}
    File.enfiler(Z)
    Tant que file non vide faire
        courant ← file.défiler()
        Pour chaque X ∈ succ(courant)
            Si X ∉ Δ
                si X est strictement interne à W alors
                        Δ ← Δ ∪{X}
                        file.enfiler(X)
                finSi
            finSi
        finPour
    finTantQue
finProcédure
```

Appliquons cet algorithme à l'exemple ci-dessus. Imaginons que l'on soit à l'étape où Δ est équivalent à l'ensemble de la partie grisée de la figure de gauche. Nous avons dans la file *X3* et *X4*. On défile *X3* et on regarde ses successeurs. Il n'y en a qu'un : *X9*. *X9* n'est pas strictement interne à *G* car *E2* en est dépendant. On n'ajoute donc pas *X9* à Δ et on n'explore pas ses successeurs. Ensuite, on défile *X4*. *X4* a deux successeurs : *X6* et *X7*. Ces deux éléments peuvent être ajoutés à Δ et on les enfile. Le processus continue de la même manière tant que la file n'est pas vide.

Critique de l'algorithme

Cet algorithme est coûteux à cause de la procédure qui recherche si *X* est strictement interne à *G*. Cette procédure consiste à parcourir Gd_B en inverse[137] jusqu'à atteindre un sommet n'appartenant pas à *G*. Si l'on atteint un tel sommet, cela signifie que *X* n'est pas strictement interne à *G*. Dans l'exemple précédent, *X8* n'est pas strictement interne à *G* car son prédécesseur *X11* n'est pas strictement interne (*E1* dépend de *X11*). Ainsi, au pire, la procédure a un temps non linéaire de l'ordre de $O(p^2)$ avec *p* le nombre d'arcs du graphe de dépendance de la grammaire *G*.

5.4.5 Un algorithme avancé de suppression de grammaires

La complexité algorithmique de la procédure naïve parcourant le graphe de dépendance de *G* (noté *Gd*) ne permet pas de rivaliser avec l'algorithme basé sur l'exploration totale du graphe de dépendance de la bibliothèque, bien que la différence de taille des graphes traités soit conséquente. L'amélioration de l'algorithme doit porter sur un point : la procédure qui détermine si un sommet *X* est strictement interne à *G*. Notre objectif est que cette procédure soit directe. Pour cela, avant de traiter tout sommet *X*, c'est à dire déterminer si on l'insère dans Δ ou non, il faut être sûr que chaque prédécesseur de *X* a déjà été traité : si on sait que chacun des prédécesseurs de *X* est strictement interne à *Gd*, *X* peut être ajouté à Δ ; autrement, on ne fait rien. Pour que cette approche soit faisable, il faut que le graphe de dépendance manipulé soit acyclique. En effet, supposons que le sommet de départ *Z* fasse partie d'un cycle et qu'il n'ait pas de prédécesseur extérieur à *Gd*. Tous ses prédécesseurs appartiennent au cycle, donc à *Gd*, et n'ont pas été traités. Par conséquent, la procédure s'arrête sans avoir fait quoi que ce soit. L'objectif est donc de travailler sur des graphes acycliques. Si l'on travaille sur le graphe *Gd* tel quel, c'est théoriquement impossible car il existe des grammaires locales strictement récursives. Une solution consiste à travailler sur le graphe condensé associé à *Gd* (noté *Gd**). Ce graphe a la propriété d'être acyclique. Mais surtout, chaque sommet de ce graphe correspond à une composante fortement connexe (CFC) de *Gd*. L'ensemble des sommets appartenant à une CFC *X** de *Gd* a une propriété fondamentale pour la suite : s'il n'existe pas de sommet extérieur à *Gd* atteignant l'un des sommets de X* alors tous les sommets de *X** peuvent être ajoutés à Δ. Dans le cas contraire, l'ajout est impossible. Ainsi, le fait de regrouper les sommets appartenant à la même CFC dans un graphe condensé et de travailler sur ce graphe revient à travailler sur le graphe *Gd*, car lorsque l'on ajoute un sommet de *Gd** dans Δ, cela revient à ajouter l'ensemble des sommets d'une CFC.
Soit *X** un sommet de *Gd**. Pour vérifier que l'on peut traiter *X**, on utilise un compteur associé à chaque sommet de *Gd** qui indique le nombre de ses prédécesseurs qui n'ont pas encore été traités. Ce compteur a été initialisé au nombre de prédécesseurs de *X**. Notons que, dans la terminologie usuelle, le nombre de prédécesseurs du sommet *X** est appelé degré entrant de *X**. Si le compteur (noté X*.compteur) n'est pas égal à 0, cela signifie que *X** possède des prédécesseurs qui n'ont pas encore été traités. On termine là le traitement de ce sommet sans explorer les sommets desquels il dépend. Par conséquent, ces derniers ne peuvent pour le moment être ajoutés à Δ[138]. Si ce compteur est égal à 0, cela signifie que tous les prédécesseurs de X* ont été traités et qu'ils sont strictement internes à *Gd* : les sommets de *X** peuvent donc être ajoutés dans Δ ; ensuite, on examine chaque successeur *Y** de *X**. On décrémente son compteur de 1 car *X** vient d'être traité et on refait récursivement le même

[137] Rappel : les arcs inverses sont également codés dans nos graphes.
[138] Si, à la fin du parcours de *Gd*, le compteur d'un sommet est toujours différent de 0, cela montre que ce sommet n'est pas strictement interne à *Gd*.

processus sur *Y**. La procédure utilisée est quasi-similaire à celle du tri topologique d'un graphe (acyclique) qui tient compte du degré entrant des sommets (R. Ahuja et al., 1993). L'algorithme est donné ci-dessous (procédure récursive EnsembleASupprimer) :

```
Fonction EnsembleASupprimer(X*,Gd*, Δ)
Résultat : un ensemble de sommets

    si X*.compteur ≠ 0 alors
        retourner Δ
    finSi
    res ← Δ ∪ X*
    pour chaque Y* ∈ X*.successeurs
        Y*.compteur ← Y*.compteur – 1
        res ← EnsembleASupprimer(Y*,Gd*,res)
    finPour
    retourner res
finFonction
```

Une fois que l'ensemble des sommets à supprimer est calculé, on peut supprimer les automates associés à chacun d'eux. Cette procédure ne pose pas de problème : mise à jour des vues par la procédure MiseAJourDesVuesBis ; suppression des symboles non-terminaux correspondants ; suppression des automates associés ; etc. L'algorithme précis est donné ci-dessous (procédure SupprimerAutomates) :

```
procédure SupprimerAutomates(Ens) //Ens : un ensemble de sommets de non-terminaux
    MiseAJourDesVuesBis(Ens)
    pour chaque X de Ens
        N_B ← N_B.supprimer(X)
        Aut_B ← Aut_B.supprimer(aut_B(X))
        si aut_B(X) est principal alors
            q0(S_B).supprimerTransition(X)
        finSi
    finPour
finProcédure
```

Enfin, la procédure principale SupprimerGrammaire fonctionne séquentiellement comme suit : phase d'initialisation, détermination de l'ensemble des automates de *B* à supprimer et suppression des automates. Dans les procédures ci-dessous, *B* et Gd_B sont considérées comme globales.

```
Procédure initialisation(X)
résultats : graphe condensé Gd* = <Vd*,Ed*> ; sommet X* de Gd*

    Gd* ← Gd_B.construireGrapheCondensé(X)
    X* ← Gd*.getCFC(X)
    pour chaque Y* ∈ Vd*
        Y*.compteur ← card(Y*.prédécesseurs) //degré entrant de Y*
    finPour
    retourner (Gd*,X*)
finProcédure

Procédure SupprimerGrammaire(X)
    (Gd*,X*) ← Initialisation(X)
    Δ ←EnsembleASupprimer(X*,Gd*, ∅)
    SupprimerAutomates(Δ)
finProcédure
```

Nous illustrons cet algorithme lorsqu'il est appliqué sur l'exemple précédent. Le graphe de dépendance de départ est le même que précédemment. Ensuite, on passe par l'étape d'initialisation consistant à calculer le graphe condensé de *Gd* et le nombre de prédécesseurs pour chacun de ses sommets. D'après la figure ci-dessous, on constate bien que *Gd** est acyclique. La première étape est très simple. Z*.compteur est égal à 0 donc on peut insérer *Z** (soit {*Z}*) dans Δ (partie grisée). Puis on parcourt ses successeurs. Son premier successeur est *X1**. On décrémente X1*.compteur de 1 : il devient nul. Puis, on traite récursivement *X1**. Son compteur est égal à 0, donc on ajoute *X1** ({*X1*}) à Δ (étape 2). On analyse ensuite son successeur *X2**. On décrémente son compteur (qui passe de 2 à 1) puis on le traite récursivement. Comme le compteur de *X2** est non nul, on retourne à l'analyse des successeurs de *Z**. Le suivant est *X2** (={*X2*,*X5*}). On décrémente son compteur de 1 et celui-ci passe à 0. Ainsi, on peut l'ajouter à Δ qui devient {*Z*, *X1*, *X2*, *X5*}. Puis, on passe à *X3** qui est aussi ajouté à Δ (étape 5). On regarde alors son successeur qui est *X9**. On soustrait 1 à son compteur qui passe à 1 (étape 6). On ne peut donc pas l'ajouter à Δ. Si l'on regarde le graphe, on constate qu'aucun sommet de *Gd* que l'on n'a pas encore traité n'atteint *X9**. Ainsi, son compteur n'atteindra jamais 0 et il ne sera jamais ajouté à Δ ainsi que les sommets desquels *X9** est dépendant (ici *X10**={*X10*}). On retourne ensuite à l'examen des successeurs de *X2**, le suivant étant *X4**. Et ainsi de suite, jusqu'à ce que le processus récursif s'arrête.

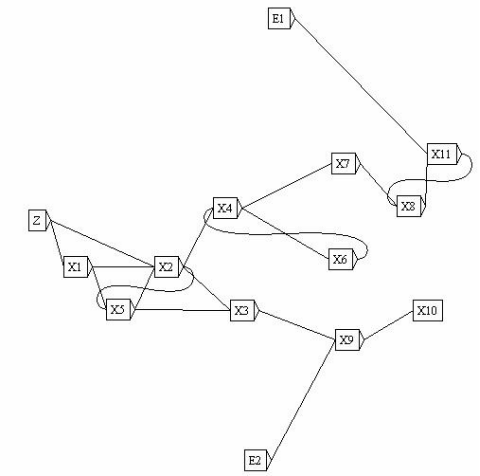

Figure 109 : graphe de départ

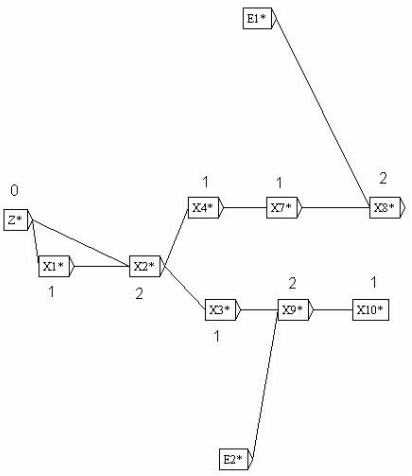

Figure 110 : graphe condensé après initialisation

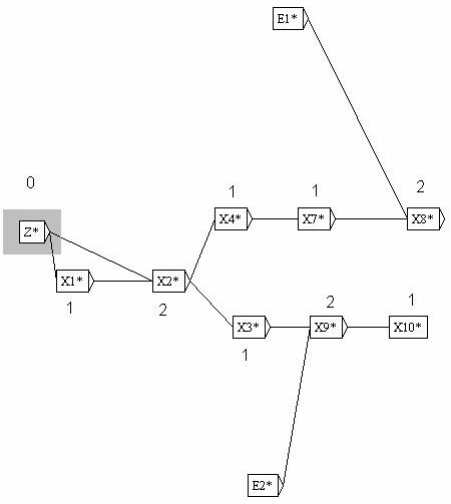

Figure 111 : 1ère étape

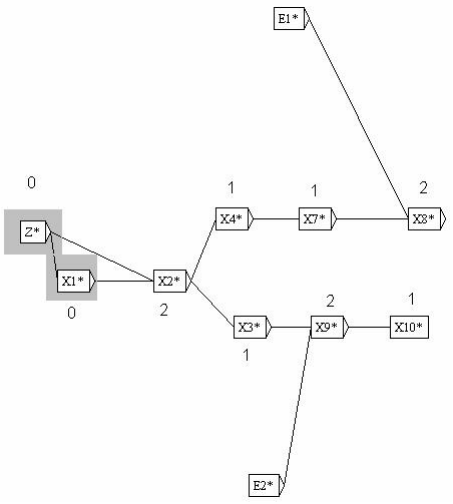

Figure 112 : 2ème étape

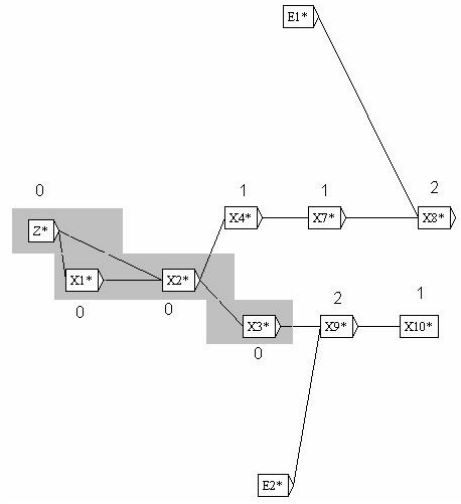

Figure 113 : 5ème étape

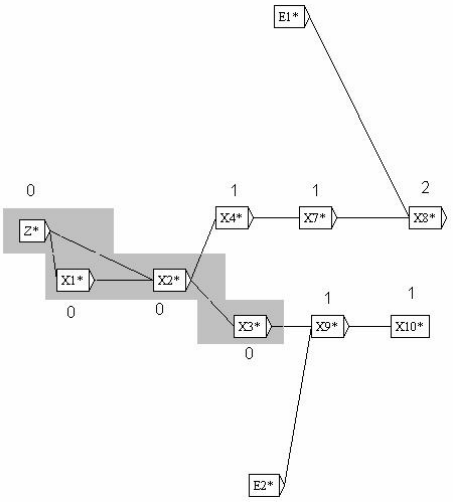

Figure 114 : 6ème étape

L'algorithme présenté s'exécute en temps linéaire car toutes les procédures mises en œuvre séquentiellement sont d'ordre *p* où *p* est le nombre d'arcs du graphe de dépendance de la grammaire à supprimer. La complexité de la procédure de calcul de l'ensemble des automates à supprimer est devenue linéaire grâce à nos améliorations. Même la procédure de construction du graphe condensé est d'ordre linéaire si l'on utilise l'algorithme de Tarjan. Ainsi, l'algorithme découlant de celui de la libération d'objets dans un programme et notre nouvel algorithme ont tous deux une complexité linéaire. Cependant, le deuxième est plus intéressant que le premier car il est censé traiter un graphe de taille relativement négligeable par rapport à l'autre (la taille de la bibliothèque versus la taille de la grammaire à supprimer).

5.5 Recherche d'information

5.5.1 Préliminaires

Vue sa taille potentielle, une bibliothèque de grammaires locales doit comporter un moteur de recherche de grammaires robuste afin d'offrir un catalogue précis de grammaires aux utilisateurs à partir d'une requête donnée. Ces requêtes peuvent porter sur des critères simples tels que l'auteur de la grammaire, la langue, le type, etc. Ces informations sont codées dans des champs classiques de bases de données relationnelles et font donc appel à des techniques très peu originales de traitement de requêtes (de type SQL par exemple). Cependant, la recherche de grammaires à partir de tels critères généraux ne présente pas forcément beaucoup d'intérêt. L'utilisateur voudra sûrement trouver des grammaires qui ont un contenu linguistique bien précis. Dans cette section, nous implantons des algorithmes de recherche de grammaires dans un ensemble de grammaires en fonction de plusieurs critères :

(a) la description de leur contenu (textes, mots-clés entrés par l'auteur)
(b) leur contenu lexical

Dans une première partie, nous implantons les algorithmes qui manipulent les critères de type (a). Nous nous inspirons de techniques classiques et avancées en recherche d'information dans des bases de données textuelles non structurées. Dans une deuxième partie, nous implémentons les algorithmes qui répondent à des critères de type (b).

5.5.2 Recherche en fonction du contenu de la documentation

5.5.2.1 Indexation des documents

La bibliothèque *B* contient un ensemble de textes *T*. Nous supposons qu'ils sont en français ou en anglais. Chacun d'entre eux documente un automate de *B* sur son contenu linguistique général (thème, type d'expressions reconnues, motivations, etc.). Lorsqu'un utilisateur cherche une grammaire, il peut très bien vouloir décrire cette grammaire de manière générale à l'aide de quelques mots clés. Ce type de recherche est équivalent à une recherche de documents dans une base de données textuelles non structurées. Nous utilisons certaines techniques de ce domaine très en vogue actuellement du fait de l'explosion de la quantité de textes sur Internet. L'indexation des documents par les mots employés est une technique classique (cf. Fluhr, 2000 pour des informations détaillées). Une première méthode traditionnelle est d'indexer les textes par mots graphiques trouvés.[139] Cette technique est triviale à implanter, surtout si l'on a un « tokeniser » à disposition (Unitex). Par exemple, soit $T = \{T_1, T_2\}$ un ensemble de 2 textes :

T_1 = Les grammaires locales aident à repérer des mots composés.
T_2 = La grammaire est une grammaire de durée.

[139] Un mot graphique = une séquence de lettres délimitée par des séparateurs.

L'indexation traditionnelle de ces textes donnerait le résultat suivant :

> {*à* (T_1), *aident* (T_1), *composés* (T_1), *de* (T_2), *des* (T_1), *durée* (T_2), *est* (T_2), *grammaire* (T_2), *grammaires* (T_1), *La* (T_2), *Les* (T_1), *locales* (T_1), *mots* (T_1), *repérer* (T_1), *une* (T_2)}

Ce découpage en mots est très basique car il ne tient pas compte des variantes flexionnelles que peuvent prendre ces mots. En effet, dans l'exemple précédent, *grammaire* et *grammaires* sont deux entrées distinctes de l'index alors qu'ils ont le même sens. Une solution avancée de plus en plus utilisée dans les moteurs de recherche est de lemmatiser les mots indexés afin d'avoir des classes de mots équivalents. Le mot désignant chaque classe est la forme canonique commune à ses membres : par exemple, *grammaire* ↔ {*grammaire*, *grammaires*}. Les fonctions externes de consultation des dictionnaires DELA implantés dans le système Unitex permettent de réaliser une telle opération. La consultation du dictionnaire de mots simples pour le texte T_1 renvoie la liste suivante où chaque ligne correspond à une entrée lexicale trouvée :

> *à,.PREP+z1*
> *aident,aider.V+z1:P3p:S3p*
> *composés,composé.A+z1:mp*
> *composés,composé.N+z1:mp*
> *composés,composer.V+z1:Kmp*
> *de,.DET+z1*
> *de,.PREP+z1*
> *de,.XI+z1*
> *grammaires,grammaire.N+z1:fp*
> *les,le.DET+z1:mp:fp*
> *les,le.PRO+z1:3mp:3fp*
> *locales,local.A+z1:fp*
> *locales,locale.N+z3:fp*
> *mots,.N+z3:mp*
> *mots,mot.N+z1:mp*
> *repérer,.V+z1:W*

Ainsi, au moyen d'une petite procédure d'analyse de cette liste, on lemmatise chaque mot du texte. Si un mot est inconnu[140], on le laisse tel quel dans l'index. On obtient alors l'index suivant pour *T* :

> {*à* (T_1), *aider* (T_1), *composé* (T_1), *composer* (T_1), *de* (T_2), *des* (T_2), *durée* (T_2), est (T_2), être (T_2), *grammaire* (T_1, T_2), *la* (T_2), *le* (T_1, T_2), *locales* (T_1), *local*(T_1), *mots* (T_1), *mot* (T_1), *repérer* (T_1), *un* (T_2)}

Le défaut majeur et bien connu de cette indexation est qu'elle peut amplifier le phénomène de l'ambiguïté naturelle. En effet, *composés* (trois fois ambigu) a été découpé en deux lemmes *composé* et *composer*. La classe associée à *composé* contient 4 mots et celle de *composer* une vingtaine. La première classe est la bonne interprétation pour *composés* dans nos textes. Ainsi, comme sa taille est plus petite que la deuxième, l'ambiguïté peut être amplifiée. Une

[140] Il n'existe pas dans le dictionnaire.

solution pour régler ce problème consiste donc à lever l'ambiguité de tels mots. Une future indexation pourrait rassembler dans une classe les mots morphologiquement et sémantiquement liés : *repérer* ↔ *repérage*, etc. Pour cela, il faut coder la liste de ces équivalences, par exemple, à partir des listes de J. Giry-Schneider (1978). Toutes ces techniques avancées sont parfaitement connues des chercheurs dans le domaine. Certains utilisent même des dictionnaires de synonymes pour agrandir les classes d'équivalences. Par ailleurs, certains mots vides comme la préposition *de*, la conjonction *et*, etc. sont extrêmement fréquents dans les textes et bruitent donc l'index. Une méthode consiste à éliminer ces mots et même des mots fréquents qui n'apportent rien comme certains déterminants (*le*), certains pronoms (*le*), etc. Certains de ces mots sont ambigus avec des mots pleins (ex : *or*), il convient donc de lever l'ambiguïté. Nos outils ne permettant pas encore de réaliser une telle opération. Pour l'instant, nous n'éliminons pas les mots vides de notre index. Il faut donc préciser à l'utilisateur d'éviter d'utiliser des mots vides dans ses requêtes. Une technique complémentaire moins connue consiste à indexer les textes selon leurs mots composés (qui forment des unités de sens). Le fait qu'une séquence de mots clés soit reconnue comme un mot composé qui se trouve dans l'index augmente la finesse de la recherche car le poids sémantique de l'unité formée par la suite des mots d'un mot composé est très fort. L'index précédent de *T* devient alors :

> *{*à *(*T_1*), aider (*T_1*), composé (*T_1*), composer (*T_1*), de (*T_2*), des (*T_2*), durée (*T_2*), est (*T_2*), être (*T_2*), grammaire (*T_1, T_2*), grammaire locale* (*T_1*), la (*T_2*), le (*T_1, T_2*), locales (*T_1*), local* (T_1), *mots (*T_1*), mot (*T_1*),mot composé (*T_1*), repérer (*T_1*), un (*T_2*)}*

La description linguistique des grammaires étant très spécialisée, il conviendra à moyen terme de construire des dictionnaires spécialisés du discours du lexique-grammaire afin de reconnaître certains termes techniques tels que *groupe nominal* ou *complément circonstanciel.*

5.5.2.2 Traitement de la requête

Avant tout traitement de la requête, les constituants simples de la séquence de mots clés doivent être lemmatisés afin de pouvoir être comparés aux entrées de l'index. Soit $u = u_1\ u_2, \ldots\ u_n$ la séquence de mots clés. Après la phase de lemmatisation, on associe à chaque mot simple u_i l'ensemble de ses lemmes U_i : *danses* → {*danse*, *danser*}. A chaque mot composé $u_i \ldots u_j$, on associe l'ensemble V_{ij} de ses lemmes. Etant donnée cet ensemble de classes (U_i et V_{ij}), il existe différents modes de recherche : OU, ET. A chacun d'eux est associé un critère de sélection du document.

- OU : il doit exister dans le texte au moins un mot simple appartenant à la classe d'un mot de $U_1 \cup U_2 \cup \ldots \cup U_n \cup \ldots \cup V_{ij} \cup \ldots$ (avec $V_{ij} \neq \varnothing$)
- ET : pour chaque U_i (resp. V_{ij}), il doit exister dans le texte un mot appartenant à la classe d'un mot de U_i (resp. V_{ij}).

Ces opérations ne posent pas de problème. Les recherches sont effectuées sur l'index. Les résultats obtenus sont ensuite classés selon deux critères :

- Premier critère : pour chaque texte, on compte le nombre de mots simples et composés de la séquence qui ont au moins un mot équivalent dans le texte (note entre 1 et $n+p$ avec p le nombre de classes de mots composés)
- Deuxième critère possible : le nombre total de mots simples et composés du texte qui ont un mot équivalent se trouvant dans u.[141]

[141] Il est aussi possible de classer les résultats suivant la proximité des mots clés dans le texte. C'est une méthode classique pour les moteurs de recherche,qui indirectement donne un poids plus important aux mots composés.

Cette méthode de classement des résultats donne un poids important aux mots composés. Cela se justifie par le fait que si un utilisateur rentre un mot composé dans sa requête, cela peut vouloir dire qu'il tient à ce que ce dernier soit présent tel quel dans le texte. Il faut donc donner un poids plus fort à un document contenant ce mot composé qu'à un document contenant l'ensemble de ses constituants mais éparpillés.

5.5.3 Recherche en fonction du contenu lexical des grammaires

5.5.3.1 Indexation par symboles terminaux

L'indexation des grammaires par symboles terminaux est aisée car ces derniers ont été normalisés lors du processus de normalisation des grammaires. Il convient d'associer chaque terminal à un ensemble d'automates où il apparaît. Pour l'instant, l'accès aux éléments terminaux n'est pas direct du fait de leur nature ensembliste et de la procédure complexe de mise en correspondance. Etant donné un symbole normalisé *x*, si l'on veut accéder à un symbole de l'alphabet terminal normalisé de la bibliothèque (T_B) qui lui correspond, il est nécessaire de parcourir la liste des terminaux. Il s'agit de comparer chaque symbole *y* de T_B avec *x* au moyen de la procédure *tagMatching(x,y)*. Dans notre application, cet accès au prix d'un parcours n'est pas un gros problème car *tagMatching* est très efficace et surtout le nombre de symboles à comparer avec les symboles terminaux de T_B est limité (moins d'une dizaine de symboles). La procédure globale est d'une complexité de *n* avec *n* le cardinal de T_B (sans doute, à terme, *n* sera égal à quelques centaines de milliers de symboles).
Un problème pourra survenir lorsque l'on voudra comparer les alphabets terminaux de deux grammaires afin de calculer leur proximité. Il conviendra alors de trouver une représentation adéquate d'ensembles de terminaux. Sachant que les symboles terminaux peuvent être représentés par une chaîne de caractère unique, il est possible pour chaque symbole *x* de T_B de construire un transducteur représentant l'ensemble des symboles qui peuvent être mis en correspondance avec *x*. En sortie du transducteur, il conviendra d'associer *x* à chacun des chemins reconnus en entrée par le transducteur. T_B serait alors l'union de ces transducteurs. Cette procédure potentielle est purement intuitive. Cette perspective mérite d'être étudiée de plus près afin de trouver une représentation optimale des ensembles de symboles terminaux linguistiques.

5.5.3.2 Définitions préliminaires

Avant de décrire les différentes procédures utilisées pour rechercher des grammaires selon leur contenu lexical, nous donnons quelques définitions. Soit la bibliothèque de grammaires $B = <N_B, T_B, aut_B, S_B>$. Soient *X* un symbole de N_B et $G = <N, T, aut, X>$ la grammaire incluse dans *B*, symbolisée par son axiome de départ *X*. On a donc $N \subseteq N_B$, $T \subseteq T_B$, $aut \subseteq aut_B$ (pour tout $X \in aut$, $aut_B(X) = aut(X)$). On dit qu'un symbole terminal *v* est contenu dans un automate *X* de *B* si *v* est une étiquette de ce dernier. On dit qu'un symbole terminal *v* est récursivement contenu dans la grammaire symbolisée par *X* (i.e. *G*) s'il existe un automate *Y* de *G*, tel que *v* en soit une étiquette.
Les procédures de recherche que nous décrivons prennent en entrée un ensemble de mots $u = \{u_1, \ldots, u_n\}$ ou une séquence ordonnée de mots $(u_1, \ldots, u_n)$. Une opération élémentaire utilisée par ces procédures est de faire correspondre chacun de ces mots avec les symboles contenus dans les automates de *B*. Il est donc nécessaire de pré-traiter *u* à l'aide des dictionnaires. Chaque u_i est une étiquette quelconque qui désigne un ensemble d'unités linguistiques. Après consultation des dictionnaires de mots simples, on peut associer un ensemble normalisé U_i à chaque u_i. Soit *x* un symbole terminal normalisé de la bibliothèque. Pour comparer u_i à *x*, il faut comparer chaque élément de U_i à *x* (procédure TagMatching). Si l'un des éléments

correspond à *x* alors u_i coïncide avec *x*. Dans le cas contraire, il n'y a pas de correspondance. Nous appelons cette procédure TagMatchingEns :

```
fonction TagMatchingEns(x,V)
    pour chaque v ∈ V
        si (TagMatching(x,v)) alors
            retourner vrai
        finSi
    finPour
    retourner faux
finFonction
```

5.5.3.3 Recherche par rapport à un sac de mots

Nous avons implanté deux procédures simples de recherche de grammaires à partir d'un sac de mots (ou symboles terminaux) $u = \{u_1, \ldots, u_n\}$ qu'a entré l'utilisateur. La première consiste à rechercher tous les automates qui contiennent au moins un mot de u[142] : c'est la procédure OU. La deuxième consiste à rechercher les grammaires qui contiennent (récursivement) tous les mots de *u* : c'est la procédure ET.
On suppose que *u* a été normalisé en $U = \{U_1, \ldots, U_n\}$. U_i est la forme normalisée de u_i. Pour chaque U_i, on regarde s'il correspond avec un symbole *t* de l'index. Si c'est le cas, u_i sélectionne les automates associés à l'entrée *t* de l'index. A la volée, nous construisons la table de hachage qui, à chaque automate sélectionné, associe l'ensemble des u_i l'ayant sélectionné. A partir de là, les deux procédures fonctionnent différemment.

- procédure OU :

La procédure OU ne requiert aucune autre opération. Il faut juste trier les grammaires sélectionnées. Nous proposons de noter chaque grammaire en fonction du nombre de u_i qui l'ont sélectionnée. Le notes vont donc de *1* à *n*. La liste des grammaires sera ordonnée en fonction de cette note : la grammaire de note la plus haute sera placée au début de la liste. Si l'utilisateur souhaite recevoir la liste de toutes les grammaires contenant récursivement au moins un symbole de *u*, il suffira de donner la liste des grammaires qui dépendent des grammaires sélectionnées en parcourant en profondeur le graphe de dépendance de *B* par ses arcs inverses.

- procédure ET :

Soit *H* la table de hachage qui pour chaque u_i associe l'ensemble des automates qu'il sélectionne. Intuitivement, l'opération suivante devrait être de réaliser l'intersection de ces ensembles. Malheureusement, le résultat obtenu contient des silences. En effet, supposons que l'on ait une grammaire qui reconnaisse des dates du type *le mardi matin, le mercredi soir,* etc. Supposons que l'automate principal *A* contienne deux sous-automates A_1 et A_2 reconnaissant respectivement *lundi, mardi, …, dimanche* et *matin, après-midi, soir,* etc. L'utilisateur entre l'ensemble *u* suivant : *soir mercredi*. Les opérations précédentes produiraient la table *H* définie comme suit : $H(soir) = \{A_2\}$; $H(mercredi) = \{A_1\}$. L'intersection des deux ensembles est vide. Donc aucune grammaire n'est sélectionnée. Or *A* contient récursivement ces deux mots, il convient donc d'améliorer notre procédure comme suit. Pour chaque u_i, il suffit d'augmenter l'ensemble $H(u_i)$ des grammaires dépendantes des grammaires de l'ensemble. Pour obtenir la liste des grammaires sélectionnées par tous les u_i, il suffit de faire

[142] En pratique, on recherche tous les automates qui contiennent une étiquette correspondant à *u*.

l'intersection de tous les ensembles de grammaires de *H*. Soit *E* l'ensemble des grammaires répondant à la requête. Alors, $E = H(u_1) \cap H(u_2) \cap \ldots \cap H(u_n)$.

5.5.3.4 Recherche par rapport à une séquence de mots

L'utilisateur peut vouloir obtenir le catalogue des grammaires qui reconnaissent une séquence qui est facteur[143] d'une séquence *u* de mots. La procédure de recherche est très simple car il suffit d'utiliser la fonction *Locate Pattern* d'Unitex (cf. S. Paumier, 2002) afin d'appliquer chaque grammaire de *B* à la séquence ordonnée *u*. Si le résultat est non vide alors on ajoute *g* au résultat de la requête. Il est possible d'effectuer au préalable un filtrage des grammaires afin de diminuer le nombre de grammaires à tester. Par exemple, si l'on avait l'arbre des préfixes de *B* de longueur inférieure à un nombre *k* donné (cf. T.A. Sudkamp, 1997)[144], on pourrait lui appliquer la séquence *u* et ainsi obtenir toutes les grammaires dont le langage possède au moins une séquence qui est facteur de longueur inférieur à *k* de *u*. Les contraintes de filtrage étant forte, la taille de l'ensemble des grammaires à tester diminuerait. Un autre filtrage peut consister à appliquer la procédure OU (définie ci-dessus) à *u*. L'efficacité de ces filtrages doit être vérifiée empiriquement.
L'utilisateur peut vouloir que la séquence *u* soit un facteur de la grammaire qu'il recherche, c'est à dire que *u* est facteur d'au moins une séquence du langage reconnu par la grammaire. Cette procédure est bien plus complexe que la précédente qui utilise des outils pré-programmés dans Unitex. On peut réaliser un premier filtrage en utilisant la procédure ET avec *u* en entrée car tous les élements de *u* doivent être contenus récursivement dans les grammaires candidates. Soit *G* l'ensemble des grammaires ayant franchi l'étape du filtrage. Pour chaque grammaire, il convient de vérifier si la séquence *u* en est un facteur. Soit *Aut* l'ensemble des automates de *B* inclus dans au moins une grammaire de *G*. *Aut* est donc l'union des ensembles de règles-automates des grammaires de *G*. Soit *Gd* le graphe de dépendance associé à *G*. On suppose $|u| > 1$.
La recherche de l'ensemble des grammaires ayant pour facteur *u* n'est pas facile. La recherche des automates ayant un facteur *u* donné est plus simple. On appelle état déclencheur d'un automate *X* tout état d'arrivée d'une transition étiquetée par u_1[145]. Vérifier qu'une séquence est facteur dans un automate consiste à trouver l'ensemble de ses états déclencheurs puis, pour chaque état *q* de cet ensemble, à reconnaître la séquence $u' = u_2 \ldots u_n$ à partir de *q*. La reconnaissance de *u'* à partir de *q* revient à vérifier s'il existe un chemin étiqueté *u'* de l'état *q* à un état quelconque de *X*. Malheureusement, il n'est pas possible d'appliquer la même procédure aux grammaires pour deux raisons :

(a) chaque automate peut posséder des sous-automates
(b) il peut-être lui-même sous-automate d'un autre automate.

Imaginons que, pour chaque automate *X* de *G*, on ait trouvé ses états déclencheurs *q* pour la séquence *u*. Nous appelons *X* l'automate de base. Nous regardons si *u* est un facteur de *X*.
Lors de la procédure de reconnaissance de *u'* à partir de *q*, si l'on rencontre un appel à un sous-automate (remarque (a)) avant d'avoir reconnu entièrement *u'*, il faut continuer la reconnaissance de *u'* dans le sous-automate en démarrant à l'état initial de ce dernier. Si l'on se trouve dans un sous-automate de l'automate de base (sous-automate direct ou non) et que l'on atteint un état final, il faut revenir à l'automate qui a appelé le sous-automate et continuer la reconnaissance de *u'* à partir de l'état d'arrivée de la transition faisant appel au sous-

[143] Soit un alphabet *T*. On dit que la séquence *u* est facteur de la séquence *v* si $v = xuy$ avec $v, u, x, y \in T^*$.
[144] Cet arbre serait un nouvel index que l'on ajouterait à la bibliothèque.
[145] Plutôt, par une étiquette correspondant à u_1.

automate. Il est donc fondamental d'avoir mémorisé dans une pile les informations nécessaires pour reprendre du bon endroit (état et automate) la reconnaissance dans un automate de niveau supérieur (ou dépendant). Si la pile est vide, cela signifie que l'automate courant est l'automate de base.
Le problème dû à (b) survient lorsque l'on atteint un état final de *X* et que *u'* n'a pas été entièrement reconnue. En effet, il est possible que le reste de la séquence *u'* soit reconnu à partir d'un état *q'* d'un automate *Y*, *q'* étant défini comme l'état d'arrivée d'une transition étiquetée par *X*. L'idée est donc de continuer la reconnaissance dans les automates qui dépendent de *X* (calculés directement à l'aide du graphe de dépendance *Gd*). Les équivalents des états déclencheurs dans les automates dépendants sont les états d'arrivée des transitions étiquetées par *X*. Par ailleurs, dès que l'on « monte » dans un automate dépendant, l'automate de base change et devient l'automate courant. En effet, *u* ne pourra plus être facteur du précédent automate de base car il n'aura reconnu la séquence *u* que partiellement.

On décrit ci-dessous le processus permettant de trouver l'ensemble *E* des grammaires de *G* acceptant *u* comme facteur. Notons que l'algorithme que nous donnons n'est pas valable dans le cas où les grammaires sont récursives à gauche. Il est donc nécessaire de vérifier au préalable cette condition sur les grammaires. Comme précédemment, on suppose que *B* et Gd_B (donc *Gd*) sont des variables globales. On utilise un ensemble de tables de hachage *W*. *W(X)* est une table de hachage associée à l'automate *X*. Ses clés sont les sous-automates directs de *X* et chaque clé *Y* est associée à l'ensemble *W(X,Y)* des états d'arrivée des transitions de *X* étiquetées par *Y*. La fonction rechercheFacteur part d'un ensemble *E* vide. Puis pour chaque automate *X* de *G*, pour chacun de ses états déclencheurs, on ajoute à *E* le résultat de la recherche dans *X* (fonction récursive rechercheFacteurRec). Cette fonction récursive prend en entrée l'état courant *q*, l'automate courant *X*, la séquence *u*, une pile *P*, l'indice *i* courant dans *u*, un ensemble *H* et les tables de hachage *W*. *H* est un ensemble de taille 1 comprenant l'automate de base. La pile *P* contient des couples (q_R,*Z*) où q_R est un état de l'automate *Z*. Si ce couple se trouve au sommet de la pile, cela signifie que l'automate courant *X* est un sous-automate de *Z* et qu'il a été appelé dans *Z* par une transition dont l'état d'arrivée est q_R. Ainsi, pour revenir à l'automate *Z* (après avoir atteint un état final de *X*), on dépile et on sait alors d'où l'on doit continuer la reconnaissance de *u'*. Inversement, si l'on passe par une transition dont l'état d'arrivée est *q'* et qui est étiquetée par un non-terminal *Y*, on empile alors le couple (*q'*,*X*) et l'on continue la reconnaissance dans le sous-automate *Y*. Si l'on passe par une transition étiquetée par un symbole terminal α et dont l'état d'arrivée est *q'*, on vérifie que u_i correspond à α (par la fonction TagMatchingEns). S'il y a correspondance, on continue la reconnaissance pour l'indice *i*+1, à partir de *q'*. Si l'on atteint la fin de la séquence *u*, *u* est un facteur de l'automate de base.

```
fonction rechercheFacteur(G,W,u)
    E ← ∅
    pour chaque automate X de G
        Déclencheurs ← Détecter les états déclencheurs de X
        pour chaque état q ∈ Déclencheurs
            P ← nouvellePile
            E ← E ∪ rechercheFacteurRec(q,X,P,u,2,{X},W)
        finPour
    finPour
    retourner E
finFonction
```

```
fonction rechercheFacteurRec(q,X,P,u,i,H,W)
    si i > longueur de u alors
        retourner H
    finSi
    res ← ∅
    si q est final dans X
        si P est vide alors //on est dans l'automate de base
            pour chaque automate Z dépendant de X
                pour chaque q' ∈ W(Z,X)
                    res ← res ∪ rechercheFacteurRec(q',Z, P, u, i, {Z}, W)
                finPour
            finPour
        sinon  //on est dans un sous-automate
            R←P.copie()
            (qR,Y) ← R.dépiler()
            res ← res ∪ rechercheFacteurRec(qR,Y, R, u, i, H, W)
        finSi
    finSi
    pour chaque transition (q,α,q') de X
        si α est terminal alors
            si MatchEtiqEns(α,ui) est vrai alors
                res ← res ∪ rechercheFacteurRec(q',X, P, u, i+1, H, W)
            finSi
        sinon // α est non-terminal
            R ← P.copie()
            R.empiler(q',X)
            res ← res ∪ rechercheFacteurRec(q0(α),α, R, u, i, H, W)
        finSi
    finPour
    retourner res
finFonction
```

Par cet algorithme, on n'obtient qu'une partie de l'ensemble *E* des grammaires de *G* qui acceptent *u* comme facteur. On complète cet ensemble en l'augmentant des grammaires de *G* qui utilisent ces grammaires (au moyen d'un parcours inverse du graphe de dépendance de *Gd*). Cette procédure est synthétisée dans la fonction suivante (fonction augmentedGrammarSet) qui renvoie l'ensemble augmenté de grammaires recherchées :

```
fonction Gd ::parcourirInverse(X)
    si X.marque est vrai alors
        retourner ∅
    finSi
    X.marque ← vrai
    res ← {X}
    pour chaque arc (Y,X) de Ed
        res ← res ∪ parcourirInverse(Y)
    finPour
    retourner res
finFonction
```

```
fonction augmentedGrammarSet(E,Gd)
    pour chaque X de Vd
        X.marque ← faux
    FinPour
    res ← ∅
    pour chaque X de E
        res ← res ∪ Gd.parcourirInverse(X)
    finPour
    retourner res
finFonction
```

5.5.4 Intersection approximative de grammaires

Un utilisateur peut être intéressé par savoir si une grammaire qu'il est en train de construire n'existe pas déjà dans la bibliothèque. La comparaison de deux grammaires revient à faire une intersection. Or il est bien connu que les langages algébriques ne sont pas fermés par intersection. Il est donc nécessaire de faire des approximations. Une première approximation consiste à transformer la grammaire locale à examiner en grammaire régulière (équivalente à un automate) car l'intersection d'un langage algébrique (la bibliothèque) et d'un langage régulier est un langage algébrique (J. Berstel, 1979).
Le formalisme des grammaires algébriques est très usuel en traitement automatique des langues. Cependant, alors que leurs tailles augmentent, le non-déterminisme des grammaires posent des problèmes d'efficacité lors de leur application à des textes. L'approximation des grammaires algébriques en automates (déterministes) est un sujet très à la mode car elle peut être extrêmement bénéfique au niveau du temps de calcul : elle supprime le problème du retour en arrière (ou « backtracking »). Les algorithmes les plus connus sont ceux de F. Pereira et R. Wright (1991,1997), M. Morhi et F. Pereira (1998) qui traitent avant tout des grammaires algébriques pondérées pour le traitement de la parole. M. Morhi et J-M. Nederhof (2001) montrent l'intérêt de d'abord transformer une grammaire algébrique en un automate sous forme compactée (système d'appel à des sous-automates). Cette forme compactée présente l'avantage d'avoir un graphe de dépendance acyclique, autorisant des traitements que le cyclisme des graphes rendait impossibles : substitution des symboles non terminaux par leurs automates, par exemple. Les automates complets peuvent donc être construits à la demande. Le processus marche comme suit. Soit *G* la grammaire à convertir. Il suffit de construire le graphe de dépendance *Gd* de *G*, puis d'en calculer le graphe condensé *G**. Chaque sommet *X** de *G** correspond à une règle non récursive ou à un ensemble de règles mutuellement récursives. Le but de l'algorithme est d'associer à chaque sommet de *X** un automate non récursif. Si l'on a une règle non récursive, on la garde telle quelle. Si l'on a un ensemble de règles mutuellement récursives, alors, pour chaque non terminal *X* associés aux sommets de *X**, on construit un automate compacté « équivalent » non récursif. Il existe différents cas : si la grammaire algébrique dont l'axiome est *X* est strictement régulière, on peut utiliser l'algorithme de dérécursivation de A.V. Aho et J.D. Ullman (1973). Si cette grammaire est strictement algébrique, on peut utiliser l'algorithme d'approximation de J-M. Nederhof (2000). La procédure de M. Mohri et J-M. Nederhof est très intéressante car elle permet de réaliser des approximations adaptées aux besoins et rend l'approximation plus réaliste. Dans le cas de notre bibliothèque, il conviendra d'adapter ce processus aux réseaux récursifs de transitions.

Nous n'avons pour l'instant pas implanté cette fonctionnalité dans notre système. Il est clair qu'elle pourrait être utile pour vérifier qu'une grammaire n'existe pas déjà dans la bibliothèque.
Cependant, il existe des problèmes. Il est d'abord nécessaire de réaliser des approximations suffisamment précises pour que le langage reconnu se rapproche le plus possible du langage

de départ ce qui est très difficile à évaluer du fait que les langages reconnus sont souvent infinis (J-M. Nederhof, 2000). Un deuxième problème est la taille mémoire (J-M. Nederhof, 2000) : une grammaire est une forme compactée ; pour la transformer en automate, il faut dupliquer les automates-règles. Nous avons réalisé une expérience démontrant que c'est un problème critique surtout avec des grammaires locales lexicalisées (M. Constant, 2001). Il conviendra d'élaguer les grammaires selon le vocabulaire avant de contruire les automates « équivalents ». Par ailleurs, comme l'intersection d'une grammaire avec un automate est coûteuse (cf. J. Berstel, 1979) et que le nombre de grammaires dans la bibliothèque est potentiellement grand, il conviendra de filtrer les grammaires selon leur vocabulaire.
L'intersection de deux grammaires n'est pas suffisante pour comparer deux grammaires. En effet, imaginons que l'on ait deux grammaires qui reconnaissent respectivement *abcde* et *abde*. Ces deux grammaires sont proches donc l'utilisateur pourrait être intéressé. Cependant, l'intersection des deux grammaires est vide. Il est donc nécessaire de réaliser des modifications. Ce problème ressemble aux recherches approximatives de motifs (recherche documentaire ou génomique), qu'il conviendra d'adapter à notre formalisme : les réseaux récursifs de transitions. Le problème reste ouvert.

5.6 Conclusion

L'avenir de l'analyse automatique des textes au moyen d'approches linguistiques passe par la construction de gigantesques ensembles de grammaires locales et une collaboration étroite entre les différents chercheurs du domaine. L'outil de gestion que nous avons implanté est donc une étape fondamentale et obligatoire. Cet outil est pour l'instant sous la forme d'un prototype testé par un nombre limité de personnes. Une des retombées immédiates de ce travail de recherche sera l'exploitation pratique de cet outil dans le réseau RELEX. A long terme, la bibliothèque constituera un état des lieux complet et détaillé des grammaires locales et pourra servir de base pour l'élaboration pratique d'outils automatiques d'analyse de textes.

Chapitre 6 : Conclusion

Les grammaires locales sous la forme de graphes ont montré leur utilité dans le domaine du TAL (M. Gross, 1997). Cependant, deux questions fondamentales se posent :

- Comment construire efficacement des grammaires précises et complètes ?
- Comment gérer le nombre et l'éparpillement de ces composants ?

Au premier problème, nous avons proposé un ensemble de méthodes. Notre démarche est avant tout empirique. Nous avons exposé des processus d'analyse linguistique et de représentation pour deux phénomènes linguistiques : expressions de mesure et adverbes de lieu contenant un nom propre locatif. Dans la directe lignée de M. Gross (1975), nous avons ramené chaque phénomène à une phrase élémentaire. Ceci nous a permis de classer sémantiquement certains phénomènes au moyen de critères formels. Par exemple, les expressions de mesure sont divisées en deux classes sémantiques : les mesures absolues (*Max a un poids de 78 kg*) et les mesures relatives (*Max est à 10 km de Luc*). Nous avons systématiquement étudié le comportement de ces phrases selon les valeurs lexicales de ses éléments. Par exemple, la distribution prépositionnelle d'un adverbe formé d'une préposition locative et d'une forme nominale composée d'un nom propre de lieu (*Méditerranée*) et de son nom classifieur associé (*mer*) dépend de la valeur lexicale du classifieur. Les faits observés ont ensuite été représentés formellement soit directement dans des graphes à l'aide d'un éditeur, soit par l'intermédiaire de tables syntaxiques ensuite converties semi-automatiquement en graphes. La méthode standard de conversion est due à E. Roche (1993). Au cours de notre travail, nous avons été confronté à des systèmes relationnels de tables syntaxiques pour lesquels la méthode standard de conversion ne fonctionnait plus. Nous avons donc élaboré une nouvelle méthode avec des formalismes et des algorithmes étendus tenant compte que les informations peuvent se trouver dans plusieurs tables.

Au deuxième problème, nous avons proposé et implanté un système de gestion de grammaires locales : une bibliothèque en-ligne de graphes. Le but est de centraliser les grammaires locales construites au sein du réseau RELEX. Nous avons conçu un ensemble d'outils permettant à la fois de stocker de nouveaux graphes et de rechercher des graphes suivant différents critères. Les systèmes traditionnels de bases de données relationnelles ne permettent pas gérer une telle bibliothèque car les composants stockés sont des objets complexes : réseaux récursifs de transitions. De ce fait, les fonctionnalités de stockage que sont l'insertion et la suppression de nouveaux graphes, ne sont pas triviales à implanter. L'implémentation d'un moteur de

recherche de grammaires nous a permis de nous pencher sur un nouveau champ d'investigation dans le domaine de la recherche d'information : la recherche d'informations linguistiques dans des grammaires locales. De nouveaux algorithmes ont donc été conçus.

Notre travail apporte une contribution à trois domaines scientifiques : linguistique, linguistique informatique et informatique. D'abord, nos études ont porté sur des champs linguistiques peu étudiés. Elles ont montré que le lexique-grammaire est une méthode très adaptée pour décrire simplement et avec précision des expressions de mesure et des adverbes locatifs. Les composants linguistiques conçus sont des nouvelles briques dans l'ensemble des descriptions formelles de la langue. Par ailleurs, les expressions numériques et les noms propres sont actuellement au centre de nombreux problèmes du domaine du TAL. Nos formalisations simples de ces phénomènes linguistiques peuvent apporter un certain nombre de réponses. Enfin, nous avons conçu une nouvelle application qui utilise des procédures complexes de mise à jour de données et de recherche d'information. L'originalité vient surtout des objets traités qui sont des composants linguistiques. Les informations ne sont plus extraites de champs classiques des bases de données mais des objets linguistiques eux-mêmes. L'implantation d'un tel outil est d'ailleurs une étape obligatoire pour une collaboration scientifique accrue et bénéfique dans le domaine de l'informatique linguistique.

Les perspectives de nos travaux sont nombreuses. La description des expressions de mesure n'est pas achevée. Il faudrait examiner les expressions indiquant une évolution de mesure et les prédicats appropriés. Par ailleurs, notre étude des adverbes locatifs contenant un nom propre de lieu géographique pourrait être prolongée pour constituer un lexique conséquent et significatif. Enfin, nous espérons que la future mise en place effective de notre bibliothèque en-ligne de graphes pourra faciliter les échanges au sein de la communauté RELEX et que, à plus long terme, elle servira de base linguistique à des outils d'analyse automatique.

Chapitre 7 : Références

Abeillé A. (1991), *Une grammaire lexicalisée d'arbres adjoints pour le français*, Thèse de doctorat, Université Paris 7

Abeillé A. (1993), *Les Nouvelles Syntaxes* (*Grammaires d'unification et analyse du français*), Collection linguistique, Armand Collin, Paris

Abeillé A., Blache P. (2000), Grammaires et analyseurs syntaxiques, In J.M. Pierrel (ed.), *Ingénierie des langues*, Hermès science, Paris

Abeillé A., Clément L., Kinyon A., Toussenel F. (2001), Un corpus français arboré : des interrogations, *Actes de la 8ème conférence sur le Traitement Automatique des Langues Naturelles*, Tours

Abney S. (1996), Statistical Methods and Linguistics, In J. Klavans, P. Resnik (eds.), *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, The MIT Press, Cambridge, MA

Agrawal R., Srikant R. (2002), Searching with numbers, *WWW2002*, Hawaii

Aho A.V., Hopcroft J.E., Ullman J.D. (1983), *Data Structures and Algorithms*, Addison-Wesley, Reading, MA

Aho A.V., Ullman J.D. (1973), *The Theory of Parsing, Translation and Compiling. Vol. II: Compiling*, Prentice-Hall, New Jersey

Ahuja R.K., Magnati T.L., Orlin J.B. (1993), *Network Flows: theory, algorithms and applications*, Prentice Hall, New Jersey

Allen J. (1995), *Natural Language Understanding*, The Benjamin/Cummings publishing Company, Redwood City, CA

Balvet A. (2000), Evaluation de stratégies linguistiques pour le filtrage d'information, In A. Dister (ed.), *Actes des Troisièmes Journées Intex*, Revue Informatique et Statistique dans les Sciences Humaines, Liège, Université de Liège

Balvet A. (2001), Filtrage d'information par analyse partielle, *Actes de la 5ème Rencontre des étudiants chercheurs en traitement automatique des langues (Récital)*, Tours

Baptista J. (1999), Manhã, Tarde, Noite: analysis of temporal adverbs using local grammars, *Seminarios de Linguistica 3*, Faro, Universidade do Algarve

Baptista J., Català D. (2002), Compound temporal adverbs in Portuguese and in Spanish, In E. Ranchhod, N. Mamede (eds.), *Advances in Natural Language Processing*, Lecture Notes in Artificial Intelligence (LNAI), 2389, Springer

Bernard P., Lecomte J., Dendien J., Pierrel J.M. (2002), Computerized linguistic resources of the research laboratory ATILF for lexical and textual analysis: Frantext, TLFi, and the software Stella, *Proceedings of the 3rd conference Language Resources and Evaluation Conference*, Las Palmas

Berstel J. (1979), *Transductions and Context-free Languages*, Teubner Verlag

Boons J.P. (1985), Préliminaires à la classification des verbes locatifs : les compléments de lieu, leurs critères, leurs valeurs aspectuelles, *Lingvisticae Investigationes*, IX:2, John Benjamins, Amsterdam

Boons J.P., Guillet A., Leclère C. (1976a), *La structure des phrases simples en français : constructions intransitives*, Droz, Genève-Paris

Boons J.P., Guillet A., Leclère C. (1976b), *La structure des phrases simples en français : classes de constructions transitives*, Rapport de Recherches, n°6, Laboratoire Documentaire et Linguistique

Borillo A. (1985), Trois jours de congé, un congé de trois jours, *Cahiers de Grammaire*, 9, Université de Toulouse - Le Mirail

Borillo A. (1989), le lexique et l'espace: les noms et les adjectifs de localisation interne, *Cahiers de grammaire*, 13, Université de Toulouse-Le-Mirail

Borillo A. (1998), *L'espace et son expression en français*, L'essentiel, Ophrys

Bresnan J., Kaplan, R.M. (1982), Lexical-Functional Grammar: A formal system for grammatical representation, In J. Bresnan (ed.), *The Mental Representation of Grammatical Relations*, The MIT Press, Cambridge, MA

Brill E. (1995), Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging, *Computational Linguistics*, 21:4

Brill E., Resnik P. (1994), A transformation-based approach to prepositional phrase attachment disambiguation, *Proceedings of COLING'94*, Kyoto, Japan

Briscoe T., Carrol J. (1993), Generalized Probabilistic LR Parsing of Natural Languge (Corpora) with Unification-Based Grammars, *Computational Linguistics*, 19:1

Briscoe T., Copestake A. (1999), Lexical rules in constraint-based grammar, *Computational Linguistics*, 25:4

Brun. C (2000), A Client/Server Architecture for Word Sense Disambiguation, Xerox

Buvet P.A. (1993), *Les déterminants nominaux quantifieurs*, Thèse de doctorat, Université Paris XIII, Villetaneuse

Buvet P.A. (1994), Les déterminants nominaux, *Lingvisticae Investigationes*, XVIII:1, John Benjamins, Amsterdam

Carvalho P., Mota C., Ranchhod E. (2002), Complex lexical units and automata, In E. Ranchhod, N. Mamede (eds.), *Advances in Natural Language Processing*, Lecture Notes in Artificial Intelligence (LNAI), 2389, Springer

Carvalho P., Ranchhod E. (2003), Analysis and Disambiguation of Nouns and Adjectives in Portuguese by FST, *Proceedings of the EACL workshop on Finite State Methods in Natural Language Processing*, Budapest

Charniak E. (1997), Statistical techniques for natural language parsing, *AI Magazine*

Chomsky N. (1957), *Syntactic Structures*, Mouton, The Hague

Christofides N. (1975), *Graph Theory: an algorithmic approach*, Academic Press, London

Chrobot A. (2000), Description des déterminants numéraux anglais par automates et transducteurs finis, In A. Dister (ed.), *Actes des Troisièmes Journées Intex*, Revue Informatique et Statistique dans les Sciences Humaines, Liège, Université de Liège

Church K. (1988), A stochastic parts program and noun phrase parser for unrestricted text, *Proceedings of the Second Conference on Applied Natural Language Processing*, Austin

Collins M., Brooks J. (1995), Prepositional Phrase Attachment through a Backed-Off Model, *Proceedings of the Third Workshop on Very Large Corpora*

Constant M. (2000), Description d'expressions numériques en français, In A. Dister (ed.), *Actes des Troisièmes Journées Intex*, Revue Informatique et Statistique dans les Sciences Humaines, Liège, Université de Liège

Constant M. (2001), Bibliothèques d'automates finis et grammaires context-free : de nouveaux traitements informatiques, *5^{e} Rencontre des étudiants chercheurs en informatique pour traitement automatique des langues (Récital)*, Tours

Constant M. (2002a), Methods for constructing lexicon-grammar resources : the example of measure expressions, *Proceedings of the 3rd conference Language Resources and Evaluation Conference*, Las Palmas

Constant M. (2002b), On the analysis of locative phrases with graphs and lexicon-grammar: the classifier/proper noun pairing, In E. Ranchhod, N. Mamede (eds), *Advances in Natural Language Processing*, Lecture Notes in Artificial Intelligence (LNAI), 2389, Springer

Constant M. (2003), Converting linguistic systems of relational matrices into Finite State Transducers, *EACL Workshop on Finite State Methods in Natural Language Processing*, Budapest

Constant M, Nakamura T., Paumier S. (2002), L'héritage des gènes MG - Localisation d'auxiliaires en français, *colloque « Grammaires et Lexiques comparés »*, Bari

Constant M, Yannacopoulou A. (2002), Le dictionnaire électronique du grec moderne : conception et développement d'outils pour son enrichissement et sa validation, *Studies in Greek Linguistics*, Proceedings of the 23rd annual meeting of the Department of Linguistics, Faculty of Philosophy, Aristote University of Thessaloniki

Copestake A., Lambeau F., Villavicencio A., Bond F., Baldwin T., Sag I., Flickinger D., (2002), Multiword Expressions: Linguistic Precision and Reusability, *Proceedings of the Third International Conference on Language Resources and Evaluation*, Las Palmas

Courtois B., Garrigues M., Gross G., Gross M., Jung R., Mathieu-Colas M., Silberztein M., Vivès R. (1997), Dictionnaire électronique des noms composés DELAC : les composants NA et NN, *Rapport Technique du LADL* n° 55, Paris, Université Paris 7

Courtois B., Silberztein M. (1990), *Les dictionnaires électroniques du français*, Langue Française 87, Larousse, Paris

Cucchiarelli, A., Luzi, D., Velardi, P. (1999), Semantic tagging of unknown proper nouns, *Natural Language Engineering*, 5:2, Cambridge University Press

Danlos L. (1980), *Représentation d'informations linguistiques : les constructions N être Prep X*, Thèse de 3e cycle, Université Paris 7

Danlos L. (1985), *Génération automatique de textes en langues naturelles*, Masson, Paris

Danlos L. (2000), Génération automatique de textes, In. J.M. Pierrel (ed.), *Ingénierie des Langues*, Hermes Science, Paris

De Négroni-Peyre D. (1978), Nominalisation par *être en* et réflexivation, *Linguisticae Investigationes*, II, John Benjamins, Amsterdam

Dermatas E., Kokkinakis G. (1995), Automatic stochastic tagging of natural language texts, *Computational Linguistics*, 21:2

Dister A. (1999), Construire des grammaires de levée d'ambiguïtés pour INTEX, In C. Fairon (ed.), *Analyse syntaxique et lexicale. Le système INTEX*, Lingvisticæ Investigationes, John Benjamins, Amsterdam

Dister A. (2000), *Actes des Troisièmes Journées Intex*, Revue Informatique et Statistique dans les Sciences Humaines, Liège, Université de Liège

Domingues C. (2001), *Etude d'outils informatiques et linguistiques pour l'aide à la recherche d'information dans un corpus documentaire*, Thèse de doctorat en informatique, Université de Marne-la-Vallée

EAGLES (1996), http://www.ilc.cnr.it/EAGLES/home.html

Eckel B. (1998), *Thinking in Java*, Prentice Hall PTR, Upper Saddle River, NJ

Fabre C., Frérot C. (2002), Groupes prépositionnels arguments ou circonstants : vers un repérage automatique en corpus, *Actes de la 9ème conférence sur le Traitement Automatique des Langues Naturelles*, Nancy

Fairon C. (1999), *Analyse lexicale et syntaxique: le système INTEX*, Lingvisticae Investigationes, John Benjamins, Amsterdam

Fairon C. (2000), *Structures non-connexes : Grammaires des incises en français : description linguistique et outils informatiques*, Thèse de doctorat en informatique, Université de Marne-la-Vallée

Fairon C. (2001), INTEX dans un système de génération automatique de tests de raisonnement analytique, http://www.nyu.edu/pages/linguistics/intex/

Fairon C., Senellart J. (1999), Classes d'expressions bilingues gérées par des transducteurs finis, dates et titres de personnalité (anglais-français), *Linguistique contrastive et traduction*, Approches empiriques, Louvain-la-Neuve

Fairon C., Watrin P. (2003), From extraction to indexation. Collecting new indexation keys by means of IE techniques, *Proceedings of the EACL workshop on Finite State Methods in Natural Language Processing*, Budapest

Ferret O., Grau B., Hurault-Plantet M., Illouz G., Jacquemin C. (2001), Utilisation des entités nommées et des variantes terminologiques dans un système de question-réponse, *Actes de la 8ème conférence sur le Traitement Automatique des Langues Naturelles*, Tours

Fluhr C. (2000), Indexation et recherche d'information textuelle, In. J.M. Pierrel (ed.), *Ingénierie des Langues*, Hermes Science, Paris

Fourour N., Morin E., Daille B. (2002), Incremental recognition and referential categorization of french proper names, *Proceedings of the Third International Conference on Language Ressources and Evaluation*, Las Palmas

Friburger N. (2002), *Reconnaissance automatique des noms propres : application à la classification automatique de textes journalistiques*, Thèse de doctorat d'informatique, Tours

Friburger N., Maurel D. (2001), Elaboration d'une cascade de transducteurs pour l'extraction de motifs : l'exemple des noms de personnes, *Actes de la 8ème conférence sur le Traitement Automatique des Langues Naturelles*, Tours

Friburger N., Dister A., Maurel D. (2000), Améliorer le découpage des phrases sous INTEX, In A. Dister (ed.), *Actes des Troisièmes Journées Intex*, Revue Informatique et Statistique dans les Sciences Humaines, Liège, Université de Liège

FSMNLP (2003), *Proceedings of the EACL workshop on Finite State Methods in Natural Language Processing*, Budapest

Gardarin G. (1999), *Bases de données: objet et relationnel*, Editions Eyrolles

Garrigues M. (1995), Prepositions and the names of countries and islands: a local grammar for the automatic analysis of texts, *Language Research*, 31:2. Seoul, Language Research Institute, Seoul National University

Garside R., Leech G., Sampson G. (1987), *The computational analysis of English: a corpus-based approach*, Longman, London

Giry-Schneider J. (1978), *Les nominalisations en français*, Droz, Genève-Paris

Giry-Schneider J. (1987), *Les prédicats nominaux en français : les phrases simples à verbe support*, Droz, Genève-Paris

Giry-Schneider J. (1991), Noms de grandeurs en avoir et noms d'unités, *Cahiers de grammaire*, Université de Toulouse-Le Mirail

Graham R.L., Knuth D. E., Patashnik O. (1994), *Concrete Mathematics: a foundation for computer science*, Addison-Wesley Publishing, second edition

Grass T., D. Maurel, O. Piton (2002), Description of a Multilingual Database of Proper Names, In E. Ranchhod, N. Mamede (eds.), *Advances in Natural Language Processing*, Lecture Notes in Artificial Intelligence (LNAI), 2389, Springer

Grevisse M. (1975), *Le bon usage*, 10e éd. rev. Gembloux : Duculot

Gross G. (1989), *Etude syntaxique de constructions converses*, Droz, Genève-Paris

Gross G. (1991), Syntaxe du complément de nom, *Linguisticae Investigationes*, XV:2, John Benjamins, Amsterdam

Gross G. (1994), Classes d'objets et description des verbes, *Langages*, 115, Larousse, Paris

Gross G. (1996), *les expressions figées en français : noms composés et autres locutions*, Ophrys

Gross M. (1975), *Méthodes en syntaxe*, Hermann, Paris

Gross M. (1977), *Grammaire transformationelle du français. Vol. 2, Syntaxe du nom*, Cantilène, Paris

Gross M. (1984), Une classification des phrases figées du français, In P. Attal, C. Muller (eds.), *De la syntaxe à la pragmatique*, Lingvisticae Investigationes Supplementa, John Benjamins, Amsterdam

Gross M. (1986), *Grammaire transformationnelle du français. Vol. 3, Syntaxe de l'adverbe.* Paris, ASSTRIL, Université Paris 7

Gross M. (1989), The Use of Finite Automata in the Lexical Representation of Natural Language, In M. Gross, D. Perrin (eds.), *Electronic Dictionaries and Automata in Computational Linguistics*, Lecture Notes in Computer Science, Springer Verlag, Berlin-New York

Gross M. (1990), Sur la notion harrissienne de transformation et son application au français, *Langages*, 99, Larousse, Paris

Gross M. (1992), Quelques réflexions sur le domaine de la traduction automatique, *TAL*, Paris

Gross M. (1996), Les formes *être Prép X* du français, *Lingvisticae Investigationes*, XX:2, John Benjamins, Amsterdam

Gross M. (1997), The Construction of Local Grammars, In E. Roche, Y. Schabes (eds.), *Finite State Language Processing*, The MIT Press, Cambridge, MA

Gross M. (1999), Lemmatization of compound tenses in English, In Fairon C. (ed), *Analyse lexicale et syntaxique: le système INTEX*, Lingvisticae Investigationes, John Benjamins publishing company, Amsterdam-Philadelphia

Gross M. (2002), Les déterminants numéraux, un exemple : les dates horaires, *Langages*, 145, Larousse, Paris

Gross M., Lentin A. (1967), *Introduction to formal grammars*, Springer-Verlag, Berlin-Heidelberg-New York

Gross M., Senellart J. (1998), Nouvelles bases statistiques pour les mots du français, *4èmes Journées Internationales d'Analyse Statistique des Données Textuelles*, Université de Nice-Sophia-Antipolis

Guillet A., Leclère C. (1992), *La structure des phrases simples en français 2 : les constructions transitives locatives*, Droz, Genève

Harris Z.S. (1951), *Methods in Structural Linguistics*, The University of Chicago Press, Chicago

Harris Z.S. (1968), *Mathematical Structures of Language*, John Wiley and sons, New York

Hindle D., Rooth R. (1994), Structural Ambiguity and Lexical Relations, *Computational Linguistics*, 19:1

Jelinek F., Lafferty J.D., Mercer R.L (1992), Basic methods of probabilistic context-free grammars, In P. Laface, R. De Mori (eds), *Speech Recognition and Understanding: Recent Advances, Trends, and Applications*, volume F75 of NATO Advanced Sciences Institute Series

Jonasson K. (1995), *Le nom propre: constructions et interprétations*, Champs linguistiques, Editions Duculot

Joshi A. (1987), Introduction to Tree Adjoining Grammar, In A. Manaster-Ramer (ed.), *The mathematics of Language*, John Benjamins, Philadelphie

Joshi A., Hopely K. (1999), A Parser From Antiquity, *Extended Finite State Models of Language*, In A. Kornai (ed.), Cambridge University Press

Karttunen L. (2001), Applications of Finite-State Transducers in Natural Language Processing, In S. Yu, A. Paun (eds.), *Implementation and Application of Automata*, Lecture Notes in Computer Science, 2088, Springer-Verlag, Heidelberg

Karttunen L. (2003), From numbers to numerals, *Proceedings of the EACL workshop on Finite State Methods in Natural Language Processing*, Budapest

Koskenniemi K. (1983), *Two level morphology : a general computational model for word-form recognition and production*, Publication 11, Department of General Linguistics, University of Helsinki

Koskenniemi K. (1990), Finite-state parsing and disambiguation, *Proceedings of COLING90*

Labelle J. (1974), *Etude de constructions avec opérateur* avoir *(nominalisations et extensions)*, Thèse de 3^{e} cycle, Université Paris 7

Lamiroy B. (1999), *Le lexique-grammaire*, Travaux de linguistique, Bruxelles

Lamiroy B. (2002), la préposition *à*, *séminaire du LADL*, Paris

Laporte E. (1988), *Méthodes algorithmiques et lexicales de phonétisation de textes : applications au français*, Thèse de doctorat en informatique, Université Paris 7

Laporte E. (1995), Appropriate nouns with obligatory modifiers, *Language Research*, 31:2, Language Research Institute, Seoul National University, Seoul

Laporte E. (2000), Mots et niveau lexical, In J.M. Pierrel (ed.), *Ingénierie des Langues*, Hermes Science, Paris

Laporte E. (2002), Le lexique-grammaire des adjectifs du français, *séminaire du LADL*, Paris

Laporte E. (2003), Applications du lexique-grammaire à l'informatique, *colloque « description linguistique pour l'analyse automatique du français »*, congrès de l'ACFAS, Rimouski

Laporte E., Monceaux A. (1999), Elimination of lexical ambiguities by grammars: the ELAG system, In C. Fairon (ed), *Analyse lexicale et syntaxique: le système INTEX*, Lingvisticae Investigationes, John Benjamins, Amsterdam

Larousse (2002), *Petit Dictionnaire Larousse*

Leclère C., Subirats-Rüggeberg, C. (1991), A bibliography of studies on lexicon-grammar, *Lingvisticæ Investigationes*, XV:2, John Benjamins, Amsterdam

Leclère C. (2002), remarque publique au *colloque "Grammaires et Lexiques comparés"*, Bari

Le Pesant D. (2002), Quelques remarques sur les constructions locatives, *colloque "Grammaires et Lexiques comparés"*, Bari

Le Pesant D. (2003), Les phrases à complément locatif : divers problèmes de syntaxe, *séminaire du LADL*, Paris

Marcus M.P., Santorini B., Marcinkiewiecz M.A. (1993), Building a large annotated corpus of English: the Penn TreeBank, *Computational Linguistics*, 19

Mathet Y. (2002), Traitement automatique des relations spatiales : un modèle des entités spatio-temporelles et de leurs relations, *La modélisation de l'espace*, Cahiers de la Maison de la Recherche en Sciences Humaines, 30

Maurel D. (1990), Adverbes de date : étude préliminaire à leur traitement automatique, *Lingvisticae Investigationes*, XIV:1, John Benjamins, Amsterdam

Maurel D., Leduc B., Courtois B. (1995), Vers la constitution d'un dictionnaire électronique des noms propres, *Lingvisticae Investigationes*, XIX:2, John Benjamins, Amsterdam

Maurel D., Piton O. (1998), Un dictionnaire de noms propres pour *INTEX* : les noms propres géographiques, In C. Fairon (ed.), *Analyse lexicale et syntaxique : le système INTEX*, Lingvisticae Investigationes, John Benjamins, Amsterdam

Mel'cuk I.A. (1988), *Dependency syntax: theory and practice*, State University Press of New York, Albany

Meunier A. (1981), *Nominalisation d'adjectifs par verbes supports*, Thèse de 3^{e} cycle, Université Paris 7

Miller G. A., Beckwith R., Fellbaum C., Gross D., Miller K.J. (1990), Introduction to WordNet: an on-line lexical database, *Journal of Lexicography*, 3

Mitkov R. (2002), Automatic Anaphora Resolution: Limits, Impediments, and Ways Forward, In E. Ranchhod, N. Mamede (eds), *Advances in Natural Language Processing*, Lecture Notes in Artificial Intelligence (LNAI), 2389, Springer

Mohri M. (1993), *Analyse et représentation par automates de structures syntaxiques composées : application aux complétives*, Thèse de doctorat en Informatique, Université Paris 7

Mohri M. (1997), Finite-State Transducers in Language and Speech Processing, *Computational Linguistics*, 23:2

Mohri M., Nederhof M-J. (2001), Regular Approximation of Context-Free Grammars through Transformation, In J.C. Junqua, G. van Noord (eds), *Robustness in Language and Speech Technology*, Kluwer Academic Publishers

Mohri M., Pereira F. (1998), Dynamic compilation of weighted context-free grammars, *Proceedings of COLING-ACL '98*, Montreal

Molinier C. (1990), Une classification des adverbes en –ment, *Langue Française*, 88, Larousse, Paris

J. Molino (1982), *Le nom propre*, Langages, 66

Nakamura, T. (à paraître), Analysing texts in a specific domain with local grammars: the case of stock exchange reports

Nakamura T., Constant M. (2001), Les expressions de pourcentage, *Flambeau*, 27, Section française de l'Université de langues étrangères de Tokyo

Nederhof M.-J. (2000), Practical experiments with regular approximation of context-free languages, *Computational Linguistics*, 26:1

Nuutila E., Soisalon-Soinninen A. (1993), On finding the strong components in a Directed Graph, *TKK report*

Oflazer K. (1996), Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction, *Computational linguistics*, 22:1

Outilex (2002), http://www.telecom.gouv.fr/rntl/AAP2001/Fiches_Resume/outilex.htm

Paumier S. (2000), Nouvelles méthodes pour la recherche d'expressions dans de grands corpus, In A. Dister (ed.), *Actes des Troisièmes Journées Intex*, Revue Informatique et Statistique dans les Sciences Humaines, Liège, Université de Liège

Paumier S. (2002), manuel d'utilisation d'Unitex, http://www-igm.univ-mlv.fr/~unitex/

Paumier S. (2003), *De la reconnaissance de formes linguistiques à l'analyse syntaxique*, Thèse de doctorat en informatique, Université de Marne-la-Vallée

Pereira F., Wright R.N. (1991), Finite-state approximation of phrase-structure grammars, *29th Annual Meeting of the Association for Computational Linguistics*, Berkeley, CA

Pereira F., Wright R.N. (1997), *Finite-state approximation of phrase-structure grammars*. In E. Roche, Y. Schabes (eds.), FiniteState Language Processing. MIT Press, Cambridge, MA

Pierrel J.M. (2000), *Ingénierie des langues*, Hermès science, Paris

Piton O., Maurel D. (1997), Le traitement informatique de la géographie politique internationale, *colloque Franche-Comté Traitement automatique des langues* (FRACTAL 97), Bulag, Besançon

Piton O., Maurel D. (2001), Les Noms Propres Géographiques et le Dictionnaire Prolintex, *Quatrièmes journées Intex*, Bordeaux

Poibeau T. (2001), Extraction d'information dans les bases de données textuelles en génomique au moyen de transducteurs à états finis, *Actes de la 8ème conférence sur le Traitement Automatique des Langues Naturelles*, Tours

Pollard C., Sag I.A. (1987), *Information-based Syntax and Semantics. Volume I. Fundamentals*, CSLI, Stanford

Pollard C., Sag I. A. (1994), *Head-Driven Phrase Structure Grammar,* U Chicago Press, Chicago

Ranchhod E. (1989), Lexique-grammaire du portugais : prédicats nominaux supportés par *estar*, *Lingvisticae Investigationes*, 13:2, John Benjamins, Amsterdam

Revuz D. (1991), *Dictionnaires et lexiques : méthode et algorithmes*, Thèse de doctorat en informatique, Université Paris 7

Roche E. (1993), Une représentation par automate fini des textes et des propriétés transformationnelles des verbes, *Lingvisticae Investigationes*, XVII:1, John Benjamins, Amsterdam

Roche E. (1993), Analyse syntaxique transformationnelle du français par transducteurs et lexique-grammaire, *Thèse de doctorat en informatique, Université Paris 7*

Roche E. (1999), Finite state transducers: parsing free and frozen sentences, In A. Kornai (ed.), *Extended finite state models of language*, Cambridge Press, MA

Roche E., Schabes Y. (1997), *Finite State Language Processing*, The MIT Press, Cambridge, MA

Romary L. (2000), Outils d'accès à des ressources linguistiques, In J.M. Pierrel (ed.), *Ingénierie des Langues*, Hermes Science, Paris

Rothenberg M. (non daté), Encore quelques remarques sur l'apposition en français

Sabah G. (2000), Sens et traitements automatiques des langues, In J.M. Pierrel (ed.), *Ingénierie des Langues*, Hermes Science, Paris

Salkoff M. (1973), *Une grammaire en chaîne du français : Analyse distributionnelle*, Dunod, Paris

Senellart J. (1998), Locating noun phrases with finite state transducers, *Proceedings of the 17th International Conference on Computational Linguistics (COLING98)*, Montréal

Senellart J. (1999a), Reconnaissance automatique des entrées du lexique-grammaire des expressions figées, In B. Lamiroy (ed.), *Le lexique-grammaire*, Travaux de linguistique, Bruxelles

Senellart J. (1999b), *Outils de reconnaissance d'expressions linguistiques complexes dans de grands corpus,* Thèse de doctorat, Université Paris 7

Sharir M. (1981), A strong-connectivity algorithm and its application in data flow analysis, *Computers and Mathematics with Applications*, 7

Silberztein M. (1993), *Dictionnaires électroniques et analyse automatique de textes : Le système INTEX*, Masson, Paris

Silberztein M. (1994), INTEX: a corpus processing system, *Proceedings of the 15th International Conference on Computational Linguistics (COLING94)*, Kyoto

Silberztein M. (1997), The Lexical Analysis of Natural Languages, In E. Roche, Y. Schabes (eds.), *Finite State Language Processing*, The MIT Press, Cambridge, MA

Silberztein M. (1999), Transducteurs pour le traitement automatique des textes, In B. Lamiroy (ed.), *Le lexique-grammaire*, Travaux de linguistique, Bruxelles

Smeaton A. (1999), Using NLP or NLP resources for Information Retrieval Tasks, In T. Strzalkowski (ed.), *Natural Language Information Retrieval*, Kluwer

Strzalkowski T., Lin F., Perez-Carballo J. (1999), Evaluating Natural Language Processing Techniques in Information Retrieval, In T. Strzalkowski (ed.), *Natural Language Information Retrieval*, Kluwer

Sudkamp T.A. (1997), *Languages and machines: an introduction to the theory of Computer Science*, second edition, Addison-Wesley

Tarjan R.E. (1972), Depth-first search and linear graph algorithms, *SIAM Journal on Computing*, 1:2

Tesnière L. (1959), *Eléments de la syntaxe structurale*, Klincksieck, Paris

Ullman J.D. (1979), *Principles of Database Systems*, Computer Science Press, Rockville, Maryland

Vandeloise C. (1985), Au-delà des descriptions géométriques et logiques de l'espace : une description fonctionnelle, *Lingvisticae Investigationes*, IX:1, John Benjamins, Amsterdam

Vandeloise C. (1986), *L'espace en français*, Seuil, Paris

Veronis J. (2000), Annotation automatique de corpus : panorama et état de la technique, In J.M. Pierrel (ed.), *Ingénierie des Langues*, Hermes Science, Paris

Vivès R. (1983), *avoir, prendre, perdre : constructions à verbe support et extensions aspectuelles*, Thèse de 3e cycle, Université Paris VIII

Voorhees E.M. (1999), Natural Language Processing and Information retrieval, *Lecture Notes in Computer Science*, Springer-Verlag

Wakao T., Gaizauskas R., Wilks Y. (1996), Evaluation of an algorithm for the recognition and classification of proper names, *Proceedings of the 16th International Conference on Computational Linguistics* (COLING96), Copenhagen

Woods W.A. (1970), Transition Network Grammars for Natural Language Analysis, *Communications of the ACM*, 13:10

Zavrel J., Daelemans W., Veenstra J. (1997), Resolving PP attachment ambiguities with memory-based learning, *Proceedings of Computational Natural Language Learning*, Madrid

Annexe

Nous fournissons en annexe quelques tables reliées à la partie consacrée aux adverbes locatifs.

- Table PNNpr
- Table NNpr-département
- Table NNpr-état-69
- Table NNpr-île
- Table NNpr-mer (extrait)
- Table NNpr-région
- Table NNpr-république

PNNpr

Loc Detc Nprc								
Loc =:à	Loc =: dans	Loc =:sur	Loc =: en	Loc =;E	Nc	Index Nc	en Nc...de ...	
+	+	+	-	-	aiguille	1	-	
-	+	+	-	-	archipel	2	-	
+	+	+	-	+	avenue	3	-	
-	+	+	+	-	baie	43	+	
+	-	+	-	+	boulevard	68	-	
-	+	+	-	-	bourg	44	-	
-	+	+	-	-	bourgade	67	-	
+	-	+	-	-	butte	45	-	
-	+	+	-	-	canton	46	-	
+	+	+	-	-	cité	47	-	cité des Ulis
-	+	+	-	-	cité	48	-	cité de Carcassone
+	+	+	-	-	col	4	-	
-	-	+	-	-	colline	49	-	
+	+	+	-	-	Commonwealth	5	-	
-	+	+	-	-	commune	50	-	
-	+	+	-	-	comté	51	-	
-	+	+	-	-	confédération	6	-	
-	+	+	-	-	Cordillère	7	-	
+	+	+	-	-	côte	52	-	côte Saint-Martin
-	-	+	-	-	côte	53	-	côte d'Azur
-	+	+	-	-	département	8	-	
-	+	+	-	-	département d'outre-mer	71	-	
-	+	+	-	-	désert	9	-	
+	+	+	-	-	émirat	10	-	
-	+	+	-	-	étang	54	-	
-	+	+	-	-	état	11	-	
-	+	+	-	-	état	69	-	
+	+	+	-	+	Etats-unis	12	-	
-	-	+	-	-	étoile	13	-	
-	+	+	-	-	fédération	14	-	
-	+	+	-	-	fleuve	15	-	
-	+	+	-	-	gave	55	-	
-	+	+	-	-	ghetto	56	-	
+	+	+	-	-	glacier	57	-	
-	+	+	-	-	golfe	58	-	
+	+	+	-	-	grand-duché	16	-	
+	+	+	-	-	île	17	-	
+	+	+	-	-	impasse	59	-	
+	+	+	-	-	lac	18	-	
+	+	+	+	-	mer	19	-	
+	+	+	-	-	mont	20	-	
-	+	+	-	-	montagne	21	-	
+	+	+	-	-	oasis	60	-	
-	+	+	-	-	océan	22	-	
-	+	+	-	-	péninsule	61	-	
+	+	+	-	-	pic	23	-	
+	-	+	-	-	place	62	-	
+	-	+	-	-	plage	24	-	
-	+	+	-	-	plaine	25	-	
+	+	+	-	-	plateau	63	-	
-	+	+	+	-	principauté	26	+	
-	+	+	-	-	protectorat	27	-	
-	+	+	-	-	province	28	-	
-	+	+	+	-	région	29	-	
-	+	+	+	-	république	30	+	

PNNpr

Loc Detc Nprc							
Loc = :à	Loc =: dans	Loc =:sur	Loc =: en	Loc =:E	Nc	Index Nc	en Nc...de ...
-	+	+	-	-	rivière	31	-
+	+	+	-	-	royaume	32	-
+	+	+	-	-	Royaume-Uni	33	-
+	+	+	-	+	rue	34	-
+	+	+	+	-	sierra	35	+
-	+	+	-	-	station balnéaire	64	-
-	+	+	-	-	station de ski	65	-
-	+	+	-	-	sultanat	36	-
-	+	+	-	-	territoire	37	-
+	-	+	-	-	tropique	38	-
-	+	+	-	-	union	39	-
-	+	+	+	-	vallée	40	+
+	+	+	-	-	village	41	-
-	+	+	+	-	ville	42	+
-	+	+	-	-	volcan	66	-

NNpr-département

Index Npr	Index Nc	Nc	Prep	Det	Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det Npr	en Npr	dans Det Npr
33	8	département	de	le	Gers	-	gersois	+	+	-	+
34	8	département	de	la	Gironde	-	girondin	+	+	+	-
35	8	département	de	l'	Hérault	-	héraultais	+	+	+	+
36	8	département	de	l'	Ille-et-Vilaine	+	-	-	+	+	+
37	8	département	de	l'	Indre	-	-	-	+	+	+
38	8	département	de	l'	Indre-et-Loire	+	-	-	+	+	+
39	8	département	de	l'	Isère	-	-	-	+	+	+
40	8	département	de	le	Jura	-	jurassien	-	+	-	+
41	8	département	de	les	Landes	-	landais	+	+	-	+
42	8	département	de	le	Loir-et-Cher	+	-	-	+	+	+
43	8	département	de	la	Loire	-	-	-	+	+	+
44	8	département	de	la	Haute-Loire	+	-	-	+	+	+
45	8	département	de	la	Loire-Atlantique	+	-	-	+	+	+
46	8	département	de	le	Loiret	-	-	-	+	-	+
47	8	département	de	le	Lot	-	lotois	+	+	-	+
48	8	département	de	le	Lot-et-Garonne	+	-	-	+	-	+
49	8	département	de	la	Lozère	+	lozérien	+	+	+	+
50	8	département	de	le	Maine-et-Loire	+	-	-	+	+	+
51	8	département	de	la	Manche	-	-	-	+	-	+
52	8	département	de	la	Marne	-	marnais	-	+	-	+
53	8	département	de	la	Haute-Marne	+	-	-	+	+	+
54	8	département	de	la	Mayenne	+	mayennais	+	+	-	+
55	8	département	de	la	Meurthe-et-Moselle	+	-	-	+	+	+
56	8	département	de	la	Meuse	-	meusien	+	+	-	+
57	8	département	de	le	Morbihan	-	morbilhannais	+	+	-	+
58	8	département	de	la	Moselle	-	mosellan	+	+	+	+
59	8	département	de	la	Nièvre	-	nivernais	+	+	-	+
60	8	département	de	le	Nord	-	-	-	+	-	+
61	8	département	de	l'	Oise	+	-	-	+	+	+
62	8	département	de	l'	Orne	-	ornais	+	+	-	+
63	8	département	de	le	Pas-de-Calais	-	-	-	+	-	+
64	8	département	de	le	Puy-de-Dôme	-	-	-	+	-	+

NNpr-département

Index Npr	Index Nc	Nc	Prep	Det	Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det Npr	en Npr	dans Det Npr
65	8	département	de	les	Pyrénées-Atlantiques	-	-	-	+	-	+
66	8	département	de	les	Hautes-Pyrénées	-	-	-	+	-	+
67	8	département	de	les	Pyrénées Orientales	-	-	-	+	-	+
68	8	département	de	le	Bas-Rhin	-	-	-	+	-	+
69	8	département	de	le	Haut-Rhin	-	-	-	+	-	+
70	8	département	de	le	Rhône-et-Loire	-	-	-	+	-	+
71	8	département	de	la	Haute-Saône	+	-	-	+	+	+
72	8	département	de	la	Saône-et-Loire	+	-	-	+	+	+
73	8	département	de	la	Sarthe	-	-	-	+	+	+
74	8	département	de	la	Savoie	+	savoyard	-	+	+	+
75	8	département	de	la	Haute-Savoie	+	-	-	+	+	+
76	8	département	de	la	Seine-Maritime	+	-	-	+	+	+
77	8	département	de	la	Seine-et-Marne	+	-	-	+	+	+
78	8	département	de	les	Yvelines	-	-	-	+	+	+
79	8	département	de	les	Deux-Sèvres	-	-	-	+	-	+
80	8	département	de	la	Somme	-	-	-	+	-	+
81	8	département	de	le	Tarn	-	-	-	+	-	+
82	8	département	de	le	Tarn-et-Garonne	+	-	-	+	-	+
83	8	département	de	le	Var	-	varois	+	+	-	+
84	8	département	de	le	Vaucluse	-	-	-	+	-	+
85	8	département	de	la	Vendée	+	vendéen	+	+	+	-
86	8	département	de	la	Vienne	-	-	-	+	+	+
87	8	département	de	la	Haute-Vienne	+	-	-	+	+	+
88	8	département	de	les	Vosges	-	-	-	+	-	+
89	8	département	de	l'	Yonne	-	icaunais	+	+	-	+
90	8	département	de	l'	Essonne	-	-	-	+	+	+
91	8	département	de	les	Hauts-de-seine	-	-	-	+	-	+
92	8	département	de	la	Seine-Saint-Denis+Seine-St-Denis	+	-	-	+	+	+
93	8	département	de	le	Val-de-Marne	+	-	-	+	+	+
94	8	département	de	le	Val-d'Oise	+	-	-	+	+	+
95	8	département	-	le	Territoire de Belfort	-	-	-	-	-	+

NNpr-état-69

Index Npr	Index Nc	Nc	Modif M2	Prep	Det	Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det	Det Npr	en Npr	dans Det Npr	à Det Npr
1	69	état	+	de	l'	Alabama	+	-	-	l'	+	+	+	-
2	69	état	+	de	l'	Alaska	+	-	-	l'	+	+	-	-
3	69	état	+	de	l'	Arizona	+	-	-	l'	+	+	+	-
4	69	état	+	de	l'	Arkansas	+	-	-	l'	+	-	+	-
5	69	état	+	de	-	Californie	+	californien	+	la	+	+	+	-
6	69	état	+	de	-	Caroline de le Nord	+	-	-	la	+	+	+	-
7	69	état	+	de	-	Caroline de le Sud	+	-	-	la	+	+	+	-
8	69	état	+	de	le	Colorado	-	-	-	le	+	-	+	+
9	69	état	+	de	le	Connecticut	-	-	-	le	+	-	+	+
10	69	état	+	de	le	Dakota de le Nord	+	-	-	le	+	+	+	+
11	69	état	+	de	le	Dakota de le Sud	+	-	-	le	+	+	+	+
12	69	état	+	de	le	Delaware	-	-	-	le	+	-	+	-
13	69	état	+	de	-	Floride	+	-	-	la	+	+	+	-
14	69	état	+	de	-	Géorgie	+	-	-	la	+	+	-	-
15	69	état	+	de	-	Hawaii	+	-	-	<E>	+	-	-	+
16	69	état	+	de	l'	Idaho	+	-	-	l'	+	+	+	-
17	69	état	+	de	l'	Illinois	+	-	-	l'	+	+	+	-
18	69	état	+	de	l'	Indiana	+	-	-	l'	+	+	+	-
19	69	état	+	de	l'	Iowa	+	-	-	l'	+	+	+	-
20	69	état	+	de	le	Kansas	-	-	-	le	+	-	+	+
21	69	état	+	de	le	Kentucky	+	-	-	le	+	-	+	+
22	69	état	+	de	-	Louisiane	+	-	-	la	+	+	+	-
23	69	état	+	de	le	Maine	-	-	-	le	+	-	+	-
24	69	état	+	de	le	Maryland	+	-	-	le	+	-	+	-
25	69	état	+	de	le	Massachussetts	-	-	-	le	+	-	+	+
26	69	état	+	de	le	Michigan	-	-	-	le	+	-	+	+
27	69	état	+	de	le	Minnesota	-	-	-	le	+	-	+	+
28	69	état	+	de	le	Mississippi	-	-	-	le	+	-	+	-
29	69	état	+	de	le	Missouri	-	-	-	le	+	-	+	-
30	69	état	+	de	le	Montana	-	-	-	le	+	-	+	+
31	69	état	+	de	le	Nebraska	-	-	-	le	+	-	+	+
32	69	état	+	de	le	Nevada	-	-	-	le	+	-	+	+
33	69	état	+	de	le	NewHampshire	-	-	-	le	+	-	+	+
34	69	état	+	de	le	NewJersey	+	-	-	le	+	-	+	+
35	69	état	+	de	-	NewYork	+	-	-	<E>	-	-	-	-
36	69	état	+	de	le	Nouveau-Mexique	-	-	-	le	+	-	-	+
37	69	état	+	de	l'	Ohio	+	-	-	l'	+	+	+	-
38	69	état	+	de	l'	Oklahoma	+	-	-	l'	+	+	+	-
39	69	état	+	de	l'	Oregon	+	-	-	l'	+	+	+	-
40	69	état	+	de	-	Pennsylvanie	+	-	-	la	+	+	+	-
41	69	état	+	de	-	Rhode Island	+	-	-	<E>	+	-	-	+
42	69	état	+	de	le	Tennessee	-	-	-	le	+	-	+	-
43	69	état	+	de	le	Texas	-	texan	+	le	+	-	+	+
44	69	état	+	de	l'	Utah	+	-	-	l'	+	+	+	-
45	69	état	+	de	le	Vermont	+	-	-	le	+	-	+	-
46	69	état	+	de	-	Virginie	+	-	-	la	+	+	+	-
47	69	état	+	de	-	Virginie-Occidentale	+	-	-	la	+	+	+	-
48	69	état	+	de	le	Washington	+	-	-	le	-	-	-	-
49	69	état	+	de	le	Wisconsin	-	-	-	le	+	+	+	+
50	69	état	+	de	le	Wyoming	-	-	-	le	+	+	+	+

NNpr-île

Index Npr	Detc pluriel	Index Nc	Nc	Modif M2	Prep	Det	Npr	LE Nc de Npr	LE Nc Npr	LE archipel de Det Npr	Npr-a	LE Nc Npr-a	Det (forme courte)	Det Npr	en Npr	dans Det Npr	à Det Npr
1	+	17	île	-	-	-	Aléoutiennes	-	+	+	-	-	les	+	-	+	+
56	-	17	île	-	de	la	Ascension	-	-	+	-	-	la	-	-	-	-
2	+	17	île	-	-	-	Bahamas	-	+	+	bahamien	+	les	+	-	+	+
3	+	17	île	+	de	les	Baléares	-	+	+	-	-	les	+	-	+	+
57	-	17	île	+	de	la	Barbade	-	-	-	barbadien	+	la	+	-	-	+
34	-	17	île	-	de	-	Beauté	+	-	-	-	-	-	-	-	-	-
4	+	17	île	-	-	-	Bermudes	-	+	+	-	-	les	+	-	+	+
35	-	17	île	+	de	-	Bornéo	+	-	-	-	-	<E>	+	-	-	+
5	+	17	île	+	de	les	Canaries	-	+	+	-	-	les	+	-	+	+
58	+	17	île	+	de	le	Cap-Vert	-	-	-	cap-verdien	+	le	+	-	-	+
6	+	17	île	-	-	-	Caraïbes	-	+	+	-	-	les	+	-	+	+
7	+	17	île	-	-	-	Célèbes	-	+	+	-	-	les	+	-	+	+
36	-	17	île	+	de	-	Ceylan	+	-	-	-	-	<E>	+	-	-	+
37	-	17	île	+	de	-	Chypre	+	-	-	chypriote	+	<E>	+	-	-	+
38	-	17	île	+	de	-	Corse	+	-	-	corse	+	la	+	+	-	-
39	-	17	île	+	de	-	Crète	+	-	-	crétois	+	la	+	+	-	-
40	-	17	île	+	de	-	Cuba	+	-	-	cubain	+	<E>	+	-	-	+
59	-	17	île	+	de	le	Diable	-	-	-	-	-	le	-	-	-	-
41	-	17	île	+	de	-	Elbe	+	-	-	-	-	<E>	+	-	-	+
8	+	17	île	-	-	-	Féroé	-	+	-	-	-	les	+	+	-	+
9	+	17	île	-	-	-	Fidji	-	+	-	fidjien	+	les	+	-	-	+
60	-	17	île	+	de	la	Grenade	-	-	-	grenadien	+	la	+	-	-	-
61	-	17	île	+	de	la	Guadeloupe	-	-	-	guadeloupéen	+	la	+	+	-	+
42	-	17	île	+	de	-	Guernesey	+	-	-	-	-	<E>	+	-	-	+
43	-	17	île	+	de	-	Haïti	+	-	-	haïtien	-	<E>	+	+	-	+
10	+	17	île	-	-	-	Hawaï	-	+	-	-	-	<E>	+	-	-	+
62	-	17	île	+	de	la	Jamaïque	-	-	-	jamaïquain	-	la	+	+	-	-
44	-	17	île	+	de	-	Java	+	-	-	-	-	<E>	+	-	-	+

NNpr-île

Index Npr	Detc pluriel	Index Nc	Nc	Modif M2	Prep	Det	Npr	LE Nc de Npr	LE Nc Npr	LE archipel de Det Npr	Npr-a	LE Nc Npr-a	Det (forme courte)	Det Npr	en Npr	dans Det Npr	à Det Npr
45 -		17 île	+		de	-	Jersey	+	-	-	-	-	<E>	+	-	-	+
12 +		17 île	-		-	-	Kerguelen	-	+	+	-	-	les	+	-	+	+
13 +		17 île	-		-	-	Kouriles	-	+	+	-	-	les	+	-	+	+
63 +		17 île	+		de	le	Levant	-	-	-	-	-	le	-	-	-	-
46 -		17 île	+		de	-	Madagascar	+	-	-	malgache	+	<E>	+	-	-	+
47 -		17 île	+		de	-	Madère	+	-	-	-	-	<E>	+	-	-	+
14 +		17 île	-		-	-	Maldives	-	+	+	maldivien	+	les	+	-	+	+
15 +		17 île	-		-	-	Malouines	-	+	+	-	-	les	+	-	+	+
48 -		17 île	+		de	-	Malte	+	-	-	maltais	+	<E>	+	-	+	+
16 +		17 île	-		-	-	Mariannes	-	+	+	-	-	les	+	-	+	+
17 +		17 île	-		-	-	Marquises	-	+	+	-	-	les	+	-	+	+
64 -		17 île	+		de	la	Martinique	-	-	-	martiniquais	+	la	+	+	-	+
19 -		17 île	-		-	-	Maurice	-	+	-	mauricien	-	<E>	-	-	-	+
70 -		17 île	+		à	les	Moines	-	-	-	-	-	les	-	-	-	-
49 -		17 île	+		de	-	Oléron	+	-	-	-	-	<E>	+	-	-	+
50 -		17 île	+		de	-	Pâques	+	-	-	-	-	-	-	-	-	-
20 +		17 île	-		-	-	Philippines	-	+	+	philippin	-	les	+	-	+	+
51 -		17 île	+		de	-	Porquerolles	+	-	-	-	-	<E>	+	-	+	+
52 -		17 île	+		de	-	Porto Rico	+	-	-	portoricain	-	<E>	+	-	-	+
53 -		17 île	+		de	-	Ré	+	-	-	-	-	-	-	-	-	-
65 -		17 île	+		de	la	Réunion	-	-	-	réunionais	-	la	+	-	-	+
54 -		17 île	+		de	-	Rhodes	+	-	-	-	-	<E>	+	-	-	+
55 -		17 île	+		de	-	Sainte-Lucie	+	-	-	saint-lucien	-	<E>	+	-	-	+
23 +		17 île	+		de	-	Saint-Pierre-et-Miquelon	+	+	-	-	-	<E>	+	-	-	+
24 -		17 île	+		de	-	Saint-Vincent-et-les-grenadines	+	-	-	saint-vincentais-et-grenadin	-	<E>	+	-	-	+
66 +		17 île	+		de	les	Samoa	-	+	+	samoan	+	les	+	-	+	+
25 -		17 île	+		de	-	Sardaigne	+	-	-	sarde	+	la	+	+	-	-
26 -		17 île	+		de	-	Sein	+	-	-	-	-	-	-	-	-	-

NNpr-île

Index Npr	Detc pluriel	Index Nc	Nc	Modif M2	Prep	Det	Npr	LE Nc de Npr	LE Nc Npr	LE archipel de Det Npr	Npr-a	LE Nc Npr-a	Det (forme courte)	Det Npr	en Npr	dans Det Npr	à Det Npr
22	+	17	île	-	-	-	Seychelles	-	+	+	seychellois	-	les	+	-	+	+
27	-	17	île	+	de	-	Sicile	+	-	-	sicilien	+	la	+	+	-	-
28	-	17	île	+	de	-	Sulawesi	+	-	-	-	-	-	+	-	-	+
29	-	17	île	+	de	-	Sumatra	+	-	-	-	-	<E>	+	-	-	+
30	-	17	île	+	de	-	Tahiti	+	-	-	tahitien	-	<E>	+	-	-	+
31	-	17	île	+	de	-	Tobago	+	-	-	-	-	<E>	+	-	-	+
67	+	17	île	+	de	les	Tonga	-	+	-	tonguien	-	les	+	-	-	+
68	-	17	île	+	de	la	Tortue	-	-	-	-	-	la	-	-	-	-
32	-	17	île	+	de	-	Touamotou	+	-	-	-	-	<E>	+	-	-	+
69	-	17	île	+	de	la	Trinité-et-Tobago	-	-	-	trinidadien	-	la	+	-	-	+
33	-	17	île	+	de	-	Yeu	+	-	-	-	-	-	-	-	-	-

NNpr-mer

Index Npr	Index Nc	Nc	Modif M2 optionnel	Prep	Det (forme longue)	Npr	Variantes Npr	LE Nc de Npr	LE Nc Npr	(Detc Nprc+Det Npr) être un Nc	Det (forme courte)	Det Npr
1	19	mer	-	-	-	Adriatique	-	-	+	+	l'	+
21	19	mer	+	de	les	Antilles	-	-	-	+	-	-
28	19	mer	+	de	les	Antilles	-	-	-	+	-	-
30	19	mer	-	de	-	Aral	-	+	-	+	-	-
2	19	mer	-	-	-	Baltique	-	-	+	+	la	+
11	19	mer	+	de	-	Barentz	-	+	-	+	-	-
3	19	mer	-	-	-	Blanche	-	-	+	+	-	-
22	19	mer	+	de	les	Caraïbes	-	-	-	+	-	-
29	19	mer	+	de	les	Caraïbes	-	-	-	+	-	-
12	19	mer	+	de	-	Chine	-	+	-	+	-	-
13	19	mer	+	de	-	Chine Méridionale	Chine de le Sud	+	-	+	-	-
14	19	mer	+	de	-	Corée	-	+	-	+	-	-
15	19	mer	+	de	-	Crète	-	+	-	+	-	-
4	19	mer	-	-	-	Egée	-	-	+	+	-	-
23	19	mer	-	de	l'	Est	-	-	-	+	-	-
16	19	mer	-	de	-	Glace	-	+	-	-	-	-
5	19	mer	-	-	-	Intérieure	-	-	+	+	-	-
17	19	mer	+	de	-	Irlande	-	+	-	+	-	-
24	19	mer	+	de	le	Japon	-	-	-	+	-	-
6	19	mer	-	-	-	Jaune	-	-	+	+	-	-
18	19	mer	+	de	-	Java	-	+	-	+	-	-
19	19	mer	+	de	-	Kara	-	+	-	+	-	-
27	19	mer	-	-	-	Manche	-	-	-	+	la	+
7	19	mer	-	-	-	Méditerranée	-	-	+	+	la	+
8	19	mer	-	-	-	Morte	-	-	+	+	-	-
9	19	mer	-	-	-	Noire	-	-	+	+	-	-
25	19	mer	-	de	le	Nord	-	-	-	+	-	-
20	19	mer	+	de	-	Norvège	-	+	-	+	-	-
10	19	mer	-	-	-	Rouge	-	-	+	+	-	-
26	19	mer	+	de	la	Tranquilité	-	-	-	+	-	-

NNpr-région

Index Npr	Index Nc	Nc	Prep	Det	Npr	LE Nc de Npr	LE Nc Npr	Npr-a	LE Nc Npr-a	Det	Det Npr	en Npr	dans Det Npr	à Det Npr
1	29	région	de	l'	Alsace	+	+	alsacien	+	l'	+	+	-	-
14	29	région	de	-	Aquitaine	+	+	aquitain	+	l'	+	+	-	-
15	29	région	de	-	Auvergne	+	+	auvergnat	+	l'	+	+	-	-
16	29	région	de	-	Basse-Normandie	+	+	-	-	la	+	+	+	-
17	29	région	de	-	Bourgogne	+	+	-	-	la	+	+	-	-
19	29	région	de	-	Bretagne	+	+	breton	+	la	+	+	-	-
2	29	région	de	le	Centre	-	+	-	-	le	+	+	+	-
20	29	région	de	-	Champagne-Ardenne	+	+	-	-	la	+	+	-	-
16	29	région	de	-	Corse	+	+	corse	+	la	+	+	+	-
22	29	région	de	-	Franche-Comté	+	+	-	-	la	+	+	+	-
23	29	région	de	-	Haute-Normandie	+	+	-	-	la	+	+	+	-
3	29	région	de	l'	Ile-de-France	+	+	francilien	+	l'	+	+	+	-
4	29	région	de	le	Languedoc-Roussillon	-	+	-	-	le	+	+	+	-
5	29	région	de	le	Limousin	-	+	limousin	+	le	+	+	+	-
6	29	région	de	la	Lorraine	-	+	lorrain	+	la	+	+	-	-
12	29	région	-	-	Midi-Pyrénées	-	+	-	-	le	+	+	+	-
7	29	région	de	le	Nord-Pas-de-Calais	-	+	-	-	le	+	+	+	-
18	29	région	-	-	PACA	-	+	-	-	le+la	+	+	+	+
8	29	région	de	les	Pays de la Loire	-	-	-	-	les	+	+	+	-
9	29	région	de	la	Picardie	+	+	picard	+	la	+	+	-	-
10	29	région	de	le	Poitou-Charentes	+	+	-	-	le	+	+	+	-
11	29	région	de	-	Provence-Alpes-Côte-de Azur	+	+	-	-	la	-	+	-	-
13	29	région	-	-	Rhône-Alpes	-	+	-	-	le	+	+	+	-

NNpr-république

Index Npr	Index Nc	Nc	Adj1	Prep	Det	Npr	LE Nc Adj1 de Det Npr	LE Nc Adj1 de Npr	LE Nc de Det Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det	Variante Det	Det Npr	Variante Npr	en Npr	dans Det Npr	à Det Npr	à Npr
106	30	république	-	-	-	-	-	-	-	-	dominicaine	+	-	-	-	-	-	-	-	-
124	30	république	socialiste	de	le	Vietnam	+	-	+	-	vietnamien	-	le	-	+	-	-	+	-	-
1	30	république	-	de	-	Afrique de le Sud	-	-	-	+	sud-africain	+	l'	-	+	-	+	-	-	-
2	30	république	-	de	-	Albanie	-	-	-	+	albanais	+	l'	-	+	-	+	-	-	-
107	30	république	fédérale	de	-	Allemagne	-	+	+	-	allemand	-	l'	-	+	-	+	-	-	-
4	30	république	-	de	-	Angola	-	-	-	+	angolais	+	l'	-	+	-	+	-	-	-
95	30	république	-	-	-	Argentine	-	-	-	-	argentin	+	l'	-	+	-	+	-	-	-
5	30	république	-	de	-	Arménie	-	-	-	+	arménien	+	l'	-	+	-	+	-	-	-
6	30	république	-	de	-	Autriche	-	-	-	+	autrichien	+	l'	-	+	-	+	-	-	-
3	30	république	-	de	-	Azerbaïdjan	-	-	-	+	-	-	l'	-	+	-	+	-	-	-
117	30	république	populaire	de	le	Bangladesh	+	-	+	-	-	-	le	-	+	-	-	-	+	-
53	30	république	-	de	le	Bélarus+Belarus	-	-	+	-	-	-	le	-	+	-	-	+	+	-
7	30	république	-	de	-	Belau	-	-	-	+	-	-	<E>	-	+	-	-	-	+	-
51	30	république	-	de	le	Bénin	-	-	+	-	-	-	le	-	+	-	-	-	+	-
52	30	république	-	de	le	Bhoutan	-	-	+	-	-	-	le	-	+	-	-	-	+	-
8	30	république	-	de	-	Biélorussie	-	-	-	+	biélorusse	+	la	-	+	-	+	-	-	-
9	30	république	-	de	-	Bolivie	-	-	-	+	bolivien	+	la	-	+	-	+	-	-	-
10	30	république	-	de	-	Bosnie-Herzégovine	-	-	-	+	bosniaque	-	la	-	+	-	+	-	-	-
54	30	république	-	de	le	Botswana	-	-	+	-	-	-	le	-	+	-	-	-	+	-
118	30	république	fédérative	de	le	Brésil	+	-	+	-	brésilien	-	le	-	+	-	-	-	+	-
11	30	république	-	de	-	Bulgarie	-	-	-	+	bulgare	+	la	-	+	-	+	-	-	-
55	30	république	-	de	le	Burundi	-	-	+	-	-	-	le	-	+	-	-	-	+	-
56	30	république	-	de	le	Cameroun	-	-	+	-	camerounais	-	le	-	+	-	-	-	+	-
57	30	république	-	de	le	Cap-Vert	-	-	+	-	cap-verdien	-	le	-	+	-	-	-	+	-
102	30	république	-	-	-	Centrafrique	-	-	-	-	centrafricaine	+	le	-	+	-	-	+	+	-
58	30	république	-	de	le	Chili	-	-	+	-	chilien	-	le	-	+	-	-	-	+	-
108	30	république	populaire	de	-	Chine	-	+	+	-	chinoise	-	la	-	+	-	+	-	-	-
12	30	république	-	de	-	Chypre	-	-	-	+	chypriote	-	<E>	-	+	-	-	-	+	-
13	30	république	-	de	-	Colombie	-	-	-	+	colombien	+	la	-	+	-	+	-	-	-

NNpr-république

Index Npr	Index Nc	Nc	Adj1	Prep	Det	Npr	LE Nc Adj1 de Det Npr	LE Nc Adj1 de Npr	LE Nc de Det Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det	Variante Det	Det Npr	Variante Npr	en Npr	dans Det Npr	à Det Npr	à Npr
59	30	république	-	de	le	Congo	-	-	+	-	congolais	-	le	-	+	-	-	-	+	-
119	30	république	démocratique	de	le	Congo	+	-	+	-	-	-	le	-	+	Zaïre	-	-	+	-
14	30	république	-	de	-	Corée	-	-	-	+	sud-coréen	-	la	-	-	Corée du Sud	+	-	-	-
60	30	république	-	de	le	Costa Rica+Costa-Rica	-	-	+	-	-	-	le	-	+	-	-	-	+	-
15	30	république	-	de	-	Côte d'Ivoire	-	-	-	+	ivoirien	-	la	-	+	-	+	-	-	-
16	30	république	-	de	-	Croatie	-	-	-	+	croate	+	la	-	+	-	+	-	-	-
17	30	république	-	de	-	Cuba	-	-	-	+	cubain	-	la	-	+	-	+	-	-	-
18	30	république	-	de	-	Djibouti	-	-	-	+	-	-	<E>	-	+	-	-	-	+	-
109	30	république	arabe	de	-	Égypte	-	+	+	-	égyptien	-	l'	-	+	-	+	-	-	-
61	30	république	-	de	l'	Équateur	-	-	+	-	équatorien	-	l'	-	+	-	+	-	-	-
19	30	république	-	de	-	Estonie	-	-	-	+	estonien	+	l'	-	+	-	+	-	-	-
62	30	république	-	de	les	Fidji	-	-	+	-	fidjien	-	les	-	+	-	-	-	+	-
20	30	république	-	de	-	Finlande	-	-	-	+	finlandais	+	la	-	+	-	+	-	-	-
96	30	république	-	-	-	France	-	-	-	-	française	+	la	-	+	-	+	-	-	-
97	30	république	-	-	-	Gabon	-	-	-	-	gabonaise	+	le	-	+	-	-	-	+	-
21	30	république	-	de	-	Gambie	-	-	-	+	-	-	la	-	+	-	+	-	-	-
22	30	république	-	de	-	Géorgie	-	-	-	+	géorgien	+	la	-	+	-	+	-	-	-
63	30	république	-	de	le	Ghana	-	-	+	-	Ghanéen	-	le	-	+	-	-	-	+	-
98	30	république	-	-	-	Grèce	-	-	-	-	hellénique	+	la	-	+	-	+	-	-	-
64	30	république	-	de	le	Guatemala	-	-	+	-	-	-	le	-	+	-	-	-	+	-
23	30	république	-	de	-	Guinée	-	-	-	+	guinéen	+	la	-	+	-	+	-	-	-
25	30	république	-	de	-	Guinée équatoriale	-	-	-	+	-	-	la	-	+	-	+	-	-	-
24	30	république	-	de	-	Guinée-Bissao+Guinée-Bissau	-	-	-	+	-	-	la	-	+	-	+	-	-	-
120	30	république	coopérative	de	le	Guyana	+	+	+	+	-	-	le	la	+	-	+	-	+	-
26	30	république	-	de	-	Haïti	-	-	-	+	haïtien	+	<E>	-	+	-	-	-	+	-
65	30	république	-	de	le	Honduras	-	-	+	-	-	-	le	-	+	-	-	-	+	-
27	30	république	-	de	-	Hongrie	-	-	-	+	hongrois	+	la	-	+	-	+	-	-	-
74	30	république	-	de	les	Îles Marshall	-	-	+	-	-	-	les	-	+	-	-	-	+	-
66	30	république	-	de	l'	Inde	-	-	+	-	indien	-	l'	-	+	-	+	-	-	-

NNpr-république

Index Npr	Index Nc	Nc	Adj1	Prep	Det	Npr	LE Nc Adj1 de Det Npr	LE Nc Adj1 de Npr	LE Nc de Det Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det	Variante Det	Det Npr	Variante Npr	en Npr	dans Det Npr	à Det Npr	à Npr
29	30 république	-	de	-	Indonésie	-	-	-	+	indonésie	+	l''	-	+	-	+	-	-	-	-
110	30 république	islamique	de	-	Iran	-	+	+	-	iranien	-	l''	-	+	-	-	-	-	-	-
28	30 république	-	de	-	Iraq+Irak	-	-	-	+	irakien	+	l''	-	+	-	+	-	-	-	-
30	30 république	-	de	-	Islande	-	-	-	+	islandais	+	l''	-	+	-	+	-	-	-	-
99	30 république	-	-	-	Italie	-	-	-	-	italienne	+	l''	-	+	-	-	+	-	-	-
67	30 république	-	de	le	Kazakhstan+Kazakstan	-	-	+	-	-	-	le	-	+	-	-	-	+	+	-
68	30 république	-	de	le	Kenya	-	-	+	-	kenyan	-	le	-	+	-	-	-	-	+	-
69	30 république	-	de	le	Kirghizistan	-	-	+	-	kirghize	+	le	-	+	-	-	-	-	+	-
31	30 république	-	de	-	Lettonie	-	-	-	+	letton	+	la	-	+	-	+	-	-	-	-
100	30 république	-	-	-	Liban	-	-	-	-	libanais	+	le	-	+	-	-	-	-	+	-
70	30 république	-	de	le	Liberia+Libéria	-	-	+	-	libérien	-	le	-	+	-	-	-	-	+	-
32	30 république	-	de	-	Lituanie	-	-	-	+	lituanien	+	la	-	+	-	+	-	-	-	-
116	30 république	yougoslave	de	-	Macédoine	-	+	+	-	macédonien	-	la	-	+	-	-	+	-	-	-
33	30 république	-	de	-	Madagascar	-	-	-	+	Malgache	+	<E>	-	+	-	-	-	-	+	-
71	30 république	-	de	le	Malawi	-	-	+	-	-	-	le	-	+	-	-	-	-	+	-
72	30 république	-	de	les	Maldives	-	-	+	-	-	-	les	-	+	-	-	-	-	+	-
73	30 république	-	de	le	Mali	-	-	+	-	malien	-	le	-	+	-	-	-	-	+	-
34	30 république	-	de	-	Malte	-	-	-	+	-	+	<E>	-	+	-	-	-	-	+	-
35	30 république	-	de	-	Maurice	-	-	-	+	-	+	<E>	-	+	-	-	-	-	+	-
111	30 république	islamique	de	-	Mauritanie	-	+	+	-	mauritanien	-	la	-	+	-	-	+	-	-	-
36	30 république	-	de	-	Moldavie+Moldova	-	-	-	+	-	-	la	-	-	-	Moldavie	+	-	-	-
75	30 république	-	de	le	Mozambique	-	-	+	-	-	-	le	-	+	-	-	-	-	+	-
37	30 république	-	de	-	Namibie	-	-	-	+	-	-	la	-	+	-	-	+	-	-	-
76	30 république	-	de	le	Népal	-	-	+	-	-	-	le	-	+	-	-	-	-	+	-
77	30 république	-	de	le	Nicaragua	-	-	+	-	-	-	le	-	+	-	-	-	-	+	-
78	30 république	-	de	le	Niger	-	-	+	-	-	-	le	-	+	-	-	-	-	+	-
121	30 république	fédérale	de	le	Nigeria+nigéria	+	-	+	-	nigérian	-	le	-	+	-	-	-	-	+	-
38	30 république	-	de	-	Niue+Nioué	-	-	-	+	-	-	<E>	-	+	-	-	-	-	+	-
79	30 république	-	de	l'	Ouganda	-	-	-	-	-	-	l''	-	+	-	-	+	-	-	-

NNpr-république

Index Npr	Index Nc	Nc	Adj1	Prep	Det	Npr	LE Nc Adj1 de Det Npr	LE Nc Adj1 de Npr	LE Nc de Det Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det	Variante Det	Det Npr	Variante Npr	en Npr	dans Det Npr	à Det Npr	à Npr
39	30	république	-	de	-	Ouzbékistan	-	-	-	+	ouzbek	+	l'	-	+	-	+	-	-	-
122	30	république	islamique	de	le	Pakistan	+	-	+	-	pakistanais	-	le	-	+	-	-	-	+	-
50	30	république	-	de	les	Palaos	-	-	+	-	-	-	les	-	+	-	-	+	+	-
94	30	république	-	de	le	Panama	-	-	+	-	panaméen	-	le	-	+	-	-	-	+	+
80	30	république	-	de	le	Paraguay	-	-	+	-	-	-	le	-	+	-	-	-	+	-
81	30	république	-	de	les	Pays-Bas	-	-	+	-	-	-	les	-	+	-	-	-	+	-
82	30	république	-	de	le	Pérou	-	-	+	-	péruvien	-	le	-	+	-	-	-	+	-
83	30	république	-	de	les	Philippines	-	-	+	-	philippin	-	les	-	+	-	-	-	+	-
40	30	république	-	de	-	Pologne	-	-	-	+	polonais	+	la	-	+	-	+	-	-	-
101	30	république	-	-	-	Portugal	-	-	-	-	portugaise	+	le	-	+	-	-	-	+	-
104	30	république	-	-	-	Rwanda	-	-	-	-	rwandaise	+	le	-	+	-	-	-	+	-
41	30	république	-	de	-	Saint-Christophe-et-Niévès	-	-	-	+	-	-	<E>	-	+	Saint-Kitts-et-Nevis	-	-	+	-
42	30	république	-	de	-	Saint-Marin+San-Marin	-	-	-	+	-	-	<E>	-	+	-	-	-	+	-
84	30	république	-	de	le+El	Salvador	-	-	+	-	-	-	le+El	-	+	-	-	-	+	-
114	30	république	démocratique	de	-	Sao Tomé-et-Principe	-	+	+	-	-	-	<E>	-	+	-	-	-	+	-
85	30	république	-	de	le	Sénégal	-	-	+	-	sénégalais	-	le	-	+	-	-	-	+	-
86	30	république	-	de	les	Seychelles	-	-	+	-	-	-	les	-	+	-	-	+	+	-
43	30	république	-	de	-	Sierra-Leone+Sierra Leone	-	-	-	+	-	-	le+la	-	+	-	+	-	+	-
44	30	république	-	de	-	Singapour	-	-	-	+	-	-	<E>	-	+	-	-	-	+	-
105	30	république	-	-	-	slovaquie	-	-	-	-	slovaque	+	la	-	+	-	+	-	-	-
45	30	république	-	de	-	Slovénie	-	-	-	+	slovène	+	la	-	+	-	+	-	-	-
115	30	république	démocratique	de	-	Somalie	-	+	+	-	somalien	-	la	-	+	-	+	-	-	-
87	30	république	-	de	le	Soudan	-	-	+	-	soudanais	-	le	-	+	-	-	-	+	-
88	30	république	-	de	le	Suriname	-	-	+	-	-	-	le	-	+	-	-	-	+	-
89	30	république	-	de	le	Tadjikistan	-	-	+	-	-	-	le	-	+	-	-	+	+	-
112	30	république	unie+-unie	de	-	Tanzanie	-	+	+	-	tanzanien	-	la	-	+	-	+	-	-	-
90	30	république	-	de	le	Tchad	-	-	+	-	tchadien	-	le	-	+	-	-	-	+	-
103	30	république	-	-	-	Tchéquie	-	-	-	-	tchèque	+	la	-	+	-	+	-	-	-
46	30	république	-	de	-	Trinité-et-Tobago	-	-	-	+	-	+	la	-	+	-	-	-	+	-

NNpr-république

Index Npr	Index Nc	Nc	Adj1	Prep	Det	Npr	LE Nc Adj1 de Det Npr	LE Nc Adj1 de Npr	LE Nc de Det Npr	LE Nc de Npr	Npr-a	LE Nc Npr-a	Det	Variante Det	Det Npr	Variante Npr	en Npr	dans Det Npr	à Det Npr	à Npr
47	30	république	-	de	-	Turquie	-	-	-	+	turque	+	la	-	+	-	+	-	-	-
123	30	république	orientale	de	l'	Uruguay	+	-	+	-	uruguayen	-	l'	-	+	-	+	-	-	-
48	30	république	-	de	-	Vanuatu	-	-	-	+	-	-	<E>	-	+	-	-	-	+	-
91	30	république	-	de	le	Venezuela	-	-	+	-	-	-	le	-	+	-	-	-	+	-
92	30	république	-	de	le	Yémen	-	-	+	-	yéménite	-	le	-	+	-	-	-	+	-
113	30	république	fédérative	de	-	Yougoslavie	-	+	+	-	yougoslave	-	la	-	+	Yougoslavie (Serbie-Monténégro)	+	-	-	-
49	30	république	-	de	-	Zambie	-	-	-	+	-	-	la	-	+	-	+	-	-	-
93	30	république	-	de	le	Zimbabwe	-	-	+	-	-	-	le	-	+	-	-	-	+	-

Index