Laboratoire d'analyse et
de technologie du langage
Université de Genève

Décembre 1997

Génération automatique de phrases

le système GBGen

Mémoire de Diplôme d'Etudes Supérieures

Auteur : Thierry Etchegoyhen

Directeur : Eric Wehrli

Contents

REMERCIEMENTS

Les remerciements ont ceci de particulier qu'il est rare que l'on accorde une part de vérité importante à leur contenu, du fait de leur caractère quasi obligatoire. Nonobstant, les remerciements qui suivent doivent être pris comme le reflet direct de mes sentiments vis-à-vis des personnes remerciées. Toute personne mettant en doute la véracité desdits remerciements sera passible de poursuites (ou de rien du tout si j'ai la flemme).

Ceci posé, je tiens à exprimer ma profonde gratitude à Eric Wehrli pour une multitude de raisons, la moindre n'étant pas d'avoir bien voulu diriger ce mémoire en me guidant dans les méandres de la linguistique informatique.

Un grand merci également à Luigi Rizzi pour sa participation au comité de soutenance et pour ses remarques incisives sur le fond et la forme, en espérant que le traitement infligé aux principes linguistiques dans ce travail ne l'aura pas trop heurté.

Parmi les gens qui ont lu attentivement ce mémoire et grandement contribué à son amélioration, je tiens à remercier particulièrement Christopher "papa" Laenzlinger (à noter qu'un autre de ses surnoms est "LaTeX"). En espérant que les *in_weak* ne l'empêchent pas de dormir.

Des remerciements groupés pour toute la bande du LATL, là encore pour un grand nombre de raisons qu'il serait superflu de détailler. Dans l'ordre alphabétique pour ne pas faire de jaloux, merci donc à : Cathy "java" Berthouzoz, Eva "Evikem" Capitao, Arnaud "bodycheck" Gaudinat, Jean-Philippe "3m20" Goldman, Patrick "chess" Jagstaidt, Juri "ex-prolog" Mengon, Lukas "oracle" Nerima, Patrick "la menace" Ruch, et Anne "oberon" Vandeventer.

Merci également à ma famille, *esker beroenak*, à mes amis de partout, et à Frédérique Tailhades pour m'avoir, à raison, fait rentrer à la maison même quand je devais finir un chapitre ou un article pour la veille dernier délai (*musu bero bero batekin*).

Il serait malhonnête de ne pas mentionner les personnes qui ont travaillé sur le projet GBGen. Sujiva Pinnagoda a élaboré une première version du prototype avant de voguer vers d'autres horizons (C++, etc.), qu'elle en soit ici remerciée. Juri Mengon travaille actuellement sur le système, en affrontant des choses aussi barbares que GB, Modula-2 ou encore un type nommé T.E., ce qui n'est pas donné à tout le monde.

Mais tous ces remerciements ne seraient rien si je ne mentionnais pas Thomas Wehrle, qui a élaboré, implémenté, conçu, le programme informatique GBGen. Je lui dois non seulement de pouvoir parler du générateur comme autre chose qu'une fantasmagorie de linguiste, mais également d'avoir appris l'essentiel de ce que je sais de la chose informatique. Qu'il veuille bien recevoir mes excuses pour les algorithmes et spécifications faibles que je lui présentais, et qu'il accueillait en secouant la tête d'un air las tout en marmonnant des choses incomprehensibles en suisse allemand (le peu que j'en comprenais avait trait à des étranglements rituels de linguistes). Merci donc pour tout, algorithmes, conseils et amitié, et, bien sûr, la prochaine tournée de flamenküchs et de bières est pour moi.

Enfin, je tiens à mentionner Georges Tsoulas, mon partenaire de recherche en linguistique et ami, qui a réussi à me fournir des commentaires pertinents dans un domaine de recherche qui n'est pas directement le sien, et tout ça pour être en bonne place dans les remerciements. Pour ce tour de force, qu'il soit également remercié.

Chapter 1

Introduction

> "Enough research will tend to support your theory."
> *Murphy's Law of Research*

Ce mémoire a essentiellement pour objectif de présenter le système de génération automatique de phrases GBGen, et les principes sous-tendant ce système. Dans cette introduction, nous ferons un survol rapide de l'état de la recherche dans le domaine de la génération et nous présenterons l'architecture du système élaboré.

Le système GBGen étant à l'heure actuelle relativement avancé dans la couverture des phénomènes tant au niveau de la représentation des connaissances que de la production syntaxique, la présentation que nous en ferons dans ce mémoire ne saurait être que partielle. Nous tâcherons cependant d'en présenter les aspects les plus pertinents au niveau de la recherche en génération automatique.

Le mémoire se divise de la façon suivante :

- La suite de ce chapitre introductif présente un bref survol de l'état de la recherche en génération, ainsi que les grandes caractéristiques du système GBGen.
- Le chapitre 2 traite de la représentation des connaissances, ou, en d'autres termes, de la définition des structures (pseudo) sémantiques qui constituent le point d'entrée ("input") du système.

- Le chapitre 3 est consacré aux questions de génération syntaxique proprement dites, de la création des structures au traitement computationnel des contraintes syntaxiques.

- Le chapitre 4 fournit un résumé des points principaux de la recherche.

1.1 L'état de la recherche en génération automatique

La génération automatique de textes (pour employer la désignation générique) est actuellement un des domaines les plus actifs en traitement automatique du langage, et ce après avoir longtemps été négligée par rapport à l'analyse automatique (cf. Zock et Sabah 1992 pour un survol de l'état de la recherche et Reiter 1994 pour l'examen de systèmes de génération représentatifs). Une des caractéristiques de nombreux travaux entrepris dans le domaine de la génération du langage reste cependant l'absence fréquente d'un modèle linguistique sous-jacent. De très nombreux systèmes utilisent encore des grammaires largement ad hoc (cf. par exemple Balicco 1993); c'est souvent le cas, en particulier, pour les modules de génération utilisés en traduction automatique. Une autre particularité courante est le caractère spécifique des systèmes de génération, très souvent liés à un domaine particulier, généralement très limité et typé (météo, rapports boursiers, etc., cf. Korelsky et Rambow 1992 pour une critique de cet état de fait). Il est frappant de voir que dans de tels systèmes, la génération syntaxique reste très dépendante des spécificités du sous-langage du domaine traité. Le problème d'une telle approche réside dans l'absence de généralité, qui nuit fortement à la réutilisation des systèmes. La difficulté d'adapter un module de génération à un autre sous-langage a du reste souvent été relevée.

On notera également que, d'une façon générale, les travaux théoriques en génération tendent à privilégier les questions relatives à la planification d'un message (le "quoi dire?", ou génération profonde) — qui relèvent plus des domaines de l'intelligence artificielle et de la psychologie que de celui de la linguistique — au détriment de la question de la réalisation linguistique des énoncés (le "comment le dire?", ou génération de surface) (cf. Zock et Sabah 1988, McDonald et Bolc 1988, Paris *et al.* 1990, Appelt 1985, Mellish & Evans 1989, Nogier 1991). Cette tendance n'est cependant plus aussi

forte à l'heure actuelle, un certain nombre de chercheurs se tournant vers les questions de génération de surface (cf. Matiasek & Trost 1996, Dorr & Gaasterland 1995, Dorr & Broman Olsen 1996, parmi d'autres).

Notons que la génération de surface elle-même a été divisée fréquemment en deux tâches. D'une part les questions de cohérence grammaticale ou de choix entre différente formes de surface, génération de phrases passives vs. actives, génération adéquate d'expressions référentielles, par exemple (cf. Danlos 1985, Dale & Reiter 1995 pour une illustration de ce type de recherche). D'autre part les questions de réalisation syntaxique, i.e. les mécanismes permettant de réaliser une phrase bien formée à partir d'informations sémantiques. Parmi les travaux de ce type, on observe une grande diversité quant au choix de la théorie. Ainsi, on trouve des travaux basés sur les grammaires systémiques (Matthiessen & Bateman 1992), les grammaires sens-texte (p.ex. Iordanskaja *et al.* 1988, 1991), les lexique-grammaires (p.ex. Danlos 1985, 1992), les grammaires d'unification (p.ex. van Noord 1990, Shieber *et al.* 1990), les TAGs (Joshi 1987, Shieber et Shabes 1990), HPSG (Matiasek & Trost 1996), ou encore l'unification fonctionnelle (Elhadad & Robin 1996).

Le modèle génératif de Chomsky, qui a servi de base à quelques travaux au début des années 70 (Friedman, 1971), n'a guère été utilisé comme base de modèles de génération syntaxique, à quelques exceptions près (Saint-Dizier 1989, Dorr 1990). Un des objectifs de ce mémoire consistera à montrer que le modèle Gouvernement et Liage de Chomsky [1981, 1986], Haegeman [1991], se révèle parfaitement adéquat comme base théorique pour un modèle de génération de phrases.[1]

Une des caractéristiques fondamentales de la recherche passée et présente en génération de surface est le peu de travaux concernant l'implémentation des *contraintes* pesant sur la formation des structures syntaxiques.[2] A titre d'exemple, il semble clair que tout système de génération (et par extension tout sytème de traduction automatique) ayant pour but de produire des énoncés à partir de représentations sémantiques doit affronter des données comme celles présentées ci-dessous :

[1] Voir également Block [1988] pour une défense du modèle de type GB en génération par rapport au modèle LFG.

[2] Ceci ne veut pas dire que des contraintes syntaxiques ne sont pas implémentées, mais simplement qu'il n'existe pas à notre connaissance de travaux rendant compte de traitements de ce type.

(1)a. Qu'a fait Marie?

b. faire(qui, quoi)

c. Qui a fait quoi?

d. *Quoi qui a fait?

La structure (1a) illustre un phénomène bien connu : la formation des interrogatives peut se faire en plaçant un élément interrogatif (élément-*wh*) en tête de phrase; dans cet exemple, c'est l'élément jouant le rôle logique de complément du verbe *faire* qui est déplacé. Tout générateur devrait, sur la base d'une structure sémantique donnée telle que (1b), produire la dérivation correcte (1c) et en aucun cas (1d) : si deux éléments-*wh* sont présents dans une structure, le sujet doit être placé en tête de phrase, de préférence à l'objet. Ce type de contrainte sur la formation des structures syntaxiques (connu sous le nom de Condition de Supériorité dans le courant chomskyen) doit nécessairement être implémenté sous peine de génération de phrases agrammaticales.

Le traitement computationnel de ce type de contraintes, qui, répétons-le, est passé sous silence dans le domaine, nous semble être l'un des points clés dans la construction d'un générateur syntaxique. Intégrer des contraintes générales sur la production des phrases permettra d'élaborer un système efficace et non lié à une application particulière ou à des langues particulières.

1.2 Buts et caractéristiques du système GB-Gen

La théorie du Gouvernement et du Liage (GB) offre un cadre adéquat pour la génération syntaxique, le modèle de grammaire proposé étant un modèle explicite de production de phrases. Une partie du travail a consisté à envisager la théorie GB sous un jour nouveau. Les considérations d'efficacité du système n'étant pas au centre du travail purement linguistique, les versions standards de la théorie ne peuvent être implémentées telles quelles. Les contraintes pesant sur les structures syntaxiques, qui, dans la théorie linguistique sont vues comme des filtres éliminant certaines dérivations, seront

revues dans l'optique d'un système déterministe.[3] Les procédures définies, qui constituent une reformulation des principes de la théorie GB, permettent d'aboutir à des dérivations bien formées, sans retour en arrière ni traitement d'alternatives.

Le système produit des structures profondes (D-structures) sur la base des informations contenues dans des structures dites pseudo-sémantiques, lesquelles constituent le point d'entrée du système de génération. S'appliquent ensuite les procédures de déplacement d'éléments, qui permettent notamment de dériver les structures interrogatives, passives, inaccusatives et à montée, donnant des S-structures grammaticalement bien formées. Les règles morphologiques de flexion ainsi que les règles de contraction ou d'élision se basent sur ces S-structures pour produire la forme de surface finale. Le flot d'information dans GBGen est décrit à la figure **??**.

Le système est relativement conforme au modèle-T de la grammaire (Chomsky 1981), à la Forme Logique près. Le modèle-T a pour caractéristiques de définir une structure profonde, sur laquelle s'applique l'instruction de mouvement *Déplacer-α* formant une S-structure, laquelle est transformée en Forme Phonétique et en Forme Logique. Comme l'illustre la figure **??**, le système ne compte pas de niveau de Forme Logique, les phénomènes propres à ce niveau étant pour l'heure laissé de côté. Notons cependant que le traitement de ces derniers n'impliquerait pas une redéfinition du système, Williams [1986] ayant montré la possiblité de fondre la Forme Logique dans la S-structure; il resterait alors à redéfinir l'analyse de Williams dans une optique déterministe. Au chapitre 3, nous justifierons le bien-fondé de l'utilisation du modèle-T pour la génération syntaxique.

L'architecture du système est décrite à la figure **??**.

TIPS et GENE sont les deux programmes (applications) principaux. Le premier est un programme basé sur l'analyseur (F)IPS (Laenzlinger & Wehrli 1991) et prend comme entrée une structure syntaxique fournie par l'analyseur, qui est ensuite analysée par le module SSAnalysis et transcrite sous forme pseudo-sémantique. Le programme GENE est interactif, les données étant rentrées par l'utilisateur et converties en structures pseudo-sémantiques,

[3]Par déterminisme nous entendons le fait pour un système de ne pas effectuer de retours en arrière ni d'élaborer plusieurs constructions en parallèle. En d'autres termes, aucune structure générée ne peut être effacée/éliminée lors du traitement.

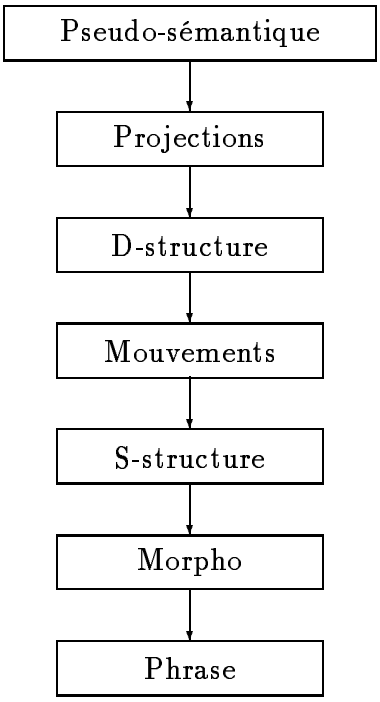


Figure 1.1: Flot d'information dans GBGen

toujours par le module SSAnalysis. Le module PSS est le module central du système et comprend i) la définition des structures de données pseudo-sémantiques, ii) les routines de conversion des structures pseudo-sémantiques en D-structures, et iii) les procédures de création des S-structures. Ces dernières sont traitées ensuite par le module MORPH qui applique les règles de conjugaison, d'accord, de contraction, etc.

Ce système très simple dans son architecture générale peut être enrichi d'une sortie vocale, en ajoutant un module de synthèse de la parole connecté au module MORPH.

Notons pour conclure cette introduction que le système est également utilisé en traduction, dans le cadre du projet CSTAR-II de traduction parole à parole dans un domaine restreint, et que le programme TIPS peut être testé sur le site web du LATL.[4] Ce mémoire permettra aux utilisateurs éventuels

[4] A l'adresse : *http://latl.unige.ch/latl/gbgen.html.*

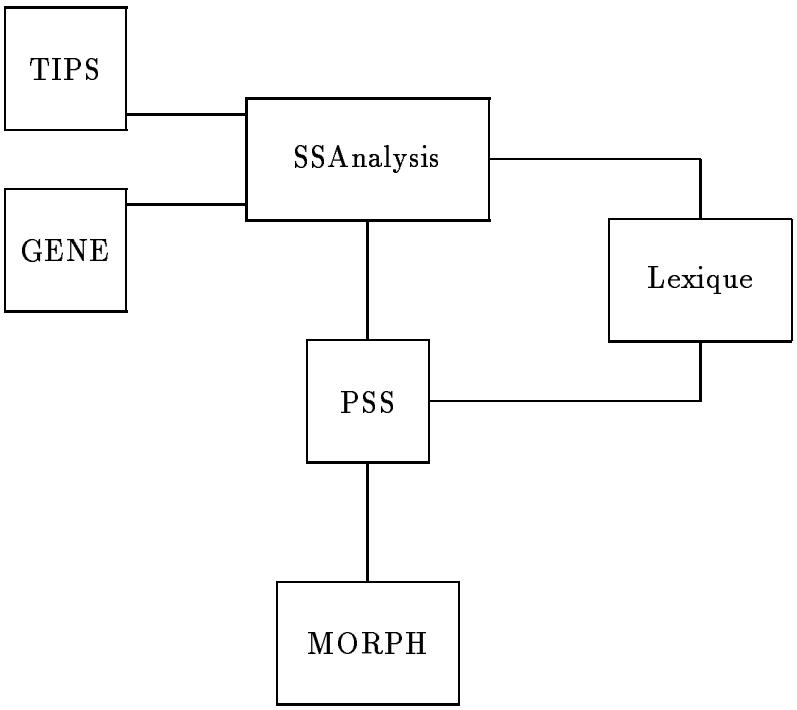


Figure 1.2: Architecture du système GBGen

du système de se faire une idée plus précise des fondements du projet ainsi que des options théoriques qui le sous-tendent.

Chapter 2

Représentation des connaissances

> "If it's not in the computer, it doesn't exist."
> *Murphy's Law of Technology*

2.1 Les structures pseudo-sémantiques : Vue générale

Tout système de génération automatique inclut, comme point d'entrée, un niveau de représentation des connaissances groupant les informations sémantiques pertinentes pour la génération de phrases. Les choix théoriques sous-tendant la définition de ce niveau sont multiples.

Dans le système GBGen, le point d'entrée est une structure dite *pseudo-sémantique*, adaptée de Clark [1993] et Clark & Wehrli [1995]. L'idée de base consiste à s'abstraire autant que possible des données relevant d'une langue particulière, sans toutefois devoir définir une représentation sémantique complète. Comme nous le verrons, ce sont en fait des structures hybrides, incluant notamment des items lexicaux tels que noms et verbes.

Notons que le fait d'inclure des items lexicaux (verbes, noms, etc.) dans ces représentations peut sembler passer outre un des problèmes fondamentaux en génération automatique, celui de la création des input sémantiques. Quelques travaux dans le domaine se sont en effet attachés à produire des

structures d'entrée n'incluant aucun item lexical. De nombreuses techniques existent, à l'état de prototype dans la plupart des cas, visant à réduire les items lexicaux sous forme de traits sémantiques plus généraux. L'interêt d'une telle approche est évident si l'on souhaite s'éloigner, au niveau de l'input, des données de langues particulières. Ce type de travail nous éloignerait cependant de la tâche de génération syntaxique proprement dite; de ce fait, nous ferons le choix d'inclure si nécessaire des items lexicaux appartenant à la langue dans laquelle on souhaite générer une phrase.[1]

Ceci posé, il reste nécessaire de s'abstraire, au niveau pseudo-sémantique, des propriétés strictement syntaxiques et également de la réalisation particulière dans une langue donnée d'une propriété sémantique. A titre d'illustration, considérons la réalisation de la datation temporelle en français et en anglais :

(2)a. J'y serai **lundi**

b. I will be there **on Monday**

Le concept temporel exprimé est le même mais la réalisation diffère dans les deux langues, l'anglais utilisant une préposition (*on*) là où le français ne présente que le nom. Un système de représentation des connaissances pour la génération se doit de ne présenter que le concept sémantique (temporel dans cet exemple) commun, et non la réalisation de ce concept dans une langue donnée.

D'un point de vue général, le principe sous-tendant la définition des représentations pseudo-sémantiques est de donner une représentation abstraite telle que :

- Les propriétés particulières des langues et les propriétés strictement syntaxiques ne soient pas indiquées,
- Le système soit gérable du point de vue computationnel.

Comme l'illustre le cas des items lexicaux mentionné précédemment, les deux conditions ci-dessus peuvent être conflictuelles : si l'abstraction désirée

[1]La décomposition en traits sémantiques des éléments lexicaux est du reste un travail qui n'a jamais été mené sur une grande ampleur : les traitements de ce type proposés dans la littérature sont basés sur de petites parties du lexique, cf. Shank [1975] pour un traitement de ce type.

entraîne une croissance du système difficilement gérable, nous trancherons en faveur de la solution la plus efficace. En d'autres termes, l'abstraction sémantique ne se fera que lorsqu'elle est souhaitable d'un point de vue computationnel. Une dernière contrainte s'applique aux représentations pseudo-sémantiques : on doit pouvoir faire correspondre à ces dernières des représentations sémantiques bien formées. Par la suite, nous donnerons quelques apercus de cette contrainte (cf. notamment le traitement des déterminants, section 2.3.2), bien qu'il ne soit pas directement pertinent dans ce travail de faire resortir le bien fondé sémantique des représentations pseudo-sémantiques.

Dans les sections suivantes, nous donnerons le détail des différents éléments composant les représentations pseudo-sémantiques. Un certain nombre de modifications seront apportées par rapport à la version implémentée actuellement; pour ne pas alourdir la présentation, nous ne signalerons pas dans tous les cas les éléments qui ont été modifiés.

2.2 Structure des représentations pseudo-sémantiques

La première étape dans la définition de nos représentations d'entrée du système consiste à définir un objet de base. Pour le définir, il est nécessaire de garder à l'esprit que le système doit pouvoir représenter des éléments non phrastiques comme ceux en (3) :[2]

(3)a. Le schmilblic.

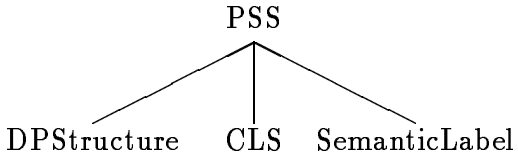
b. Vers la montagne.

c. Elle.

D'autre part, il est évidemment nécessaire de pouvoir représenter des phrases. Nous ferons donc du type PSS (pour "pseudo-semantic structure"), défini ci-dessous, le type central des représentations pseudo-sémantiques :

[2]Ceci est essentiel pour l'utilisation du générateur en traduction. De nombreux textes présentent par exemple des titres qui ne sont que des groupes nominaux, prépositionnels, etc., et non des phrases complètes.

(4)



Une PSS peut donc être un type *DPStructure*, représentant les groupes nominaux, un type *CLS* ("clause structure") pour les phrases, ou encore un type *SemanticLabel* pour les éléments ayant une fonction sémantique particulière mais ne portant pas de rôle thématique (cf. la section 2.3.3).

Ainsi, un groupe nominal, une phrase ou un label définissent les sous-types de base du type PSS, lequel est l'élément de base des représentations. Comme nous le verrons par la suite, chacune de ces sous-unités peut contenir les autres, de façon à représenter la récursivité des structures langagières.

2.3 Détail des types PSS

2.3.1 PSS I : CLS

Modes de jugement

Dans la définition pseudo-sémantique d'une phrase il est nécessaire d'intégrer des notions générales permettant de distinguer de grands types de constructions. Il est ainsi nécessaire, par exemple, de pouvoir distinguer une construction impersonnelle (5a) de sa contrepartie plus "classique", prédicative, (5b) :

(5)a. Il est venu trois personnes.

b. Trois personnes sont venues.

Pour traiter cette distinction, nous introduirons la distinction classique du philosophe Brentano, reprise par de nombreux auteurs en linguistique (Kuroda 1972, Ladusaw 1994), entre jugements de types Thetic et Categorical. Plus précisément, nous suivrons ici l'analyse proposée dans Etchegoyhen & Tsoulas [1997], groupant les impersonnelles dans la classe Thetic et les phrases de type Sujet-Prédicat dans la classe Categorical. D'un point de vue

général, la classe Thetic regroupe toutes les constructions présentatives, qui constituent d'un point de vue sémantique des descriptions (d'objets, d'individus, d'événements), et la classe Categorical les phrases à prédication classique où un sujet est présenté, sur lequel s'applique la prédication (verbale, dans le cas habituel). Cette distinction est bien évidemment insuffisante pour pouvoir discriminer toutes les variantes de phrase. Ainsi, toutes les impersonnelles ne sont pas semblables d'un point de vue sémantique. Considérons, à titre d'exemple, la phrase suivante :

(6) C'est un policier.

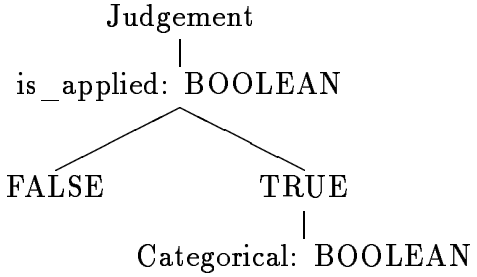
Cette construction peut-être interprétée de deux façons, A- comme une description d'un objet du monde, que l'on appelle "policier", la phrase répondrait donc à une question du type *Qu'est-ce que c'est?*, ou B- comme la réponse à une question du type *Qui a décleché l'émeute?*. La distinction n'est pas superflue pour un générateur syntaxique, des effets de définitude, donc de production du déterminant correct, distinguant les deux constructions. Ainsi, pour une impersonnelle de type A, la phrase (7) serait incorrecte :

(7) A- *C'est le policier.

La phrase ci-dessus ne peut en aucun cas constituer une description d'un objet (cf. Etchegoyhen & Tsoulas 1997 pour une analyse du phénomène), bien que la phrase soit parfaitement correcte en tant que phrase de type B. Il apparaît donc nécessaire d'avoir une caractérisation plus raffinée dans le système GBGen, de ce genre de phénomènes, de façon à pouvoir appliquer les contraintes appropriées dans le processus de génération. Pour traiter ces cas, nous opterons pour la structure suivante :[3]

[3] *Categorical* et *is_applied* sont des variables booléennes, i.e. pouvant prendre les valeurs TRUE ou FALSE uniquement.

(8)



Les phrases pour lesquelles le Type Judgement n'est pas pertinent auront la valeur FALSE pour la variable *is_ applied*. Pour l'heure nous considérerons que ce cas de figure correspond aux phrases existentielles, telles que (9) :

(9) Il y avait Marie (à la soirée)

Il se peut que ces phrases fassent partie du mode de jugement Thetic, décrivant une situation, nous laisserons cependant cette question de côté pour l'instant.

Les phrases associées à un jugement Categorical (cf. (4b) auront la valeur TRUE pour la variable *categorical*; enfin, les phrases du type (4a) auront la valeur FALSE pour la variable *categorical* (ce qui équivaut à leur associer la valeur Thetic).

Type énonciatif

Le type énonciatif (Utterance type) de la phrase à générer doit également être indiqué dans toute CLS. Les phrases de type interrogatif ou exclamatif déclenchent par exemple le mouvement *wh* en syntaxe, contrairement aux phrases déclaratives. Les phrases suivantes illustrent ce type de mouvement:

(10)a. Par qui est-ce que l'assistant a été frappé?

b. Le professeur se demande quel assistant acheter.

c. Quel bel assistant vous avez acheté!

Les options supplémentaires à inclure dans les CLS sont donc les suivantes :

(11) Utterance type :

1. Déclaratif
2. Interrogatif
3. Exclamatif

Notons que les phrases clivées ne résulteront pas d'un paramètre [± clivée], mais d'un trait de focalisation associé à la structure focalisée. Ce trait entraîne une structure clivée en français et en anglais, mais pas dans d'autres langues (e.g. allemand, basque, parmi d'autres), comme illustré ci-dessous :

(12)a. C'est un homme que Marie a vu

b. * Es ist einen Mann daß Maria gesehen hat

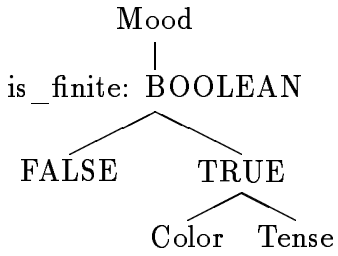
c. EINEN MANN hat Maria gesehen

La représentation pseudo-sémantique sera la même pour le français et l'allemand quant au trait [+focus] associé aux DPs *un homme/einen Mann*, mais la réalisation syntaxique sera différente, du fait des propriétés particulières de chacune de ces langues en ce qui concerne la réalisation de la focalisation.

Modes

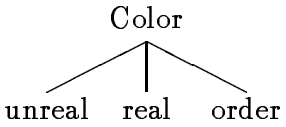
Toute CLS doit également contenir les informations de mode pertinentes pour la génération d'une phrase. Il est important dans un premier temps de pouvoir distinguer les phrases tensées des non-tensées. Pour ce faire, nous définissons un type Mood comme suit :

(13)



La valeur FALSE de la variable *is_finite* donne les constructions infinitives et gérondives (qui se distinguent par l'Aspect, voir plus bas); la valeur TRUE englobe les constructions tensées et met en jeu les types Color et Tense. Ce dernier sera défini en détail au paragraphe suivant. Quant au type Color, il sera défini comme ci-dessous :

(14)

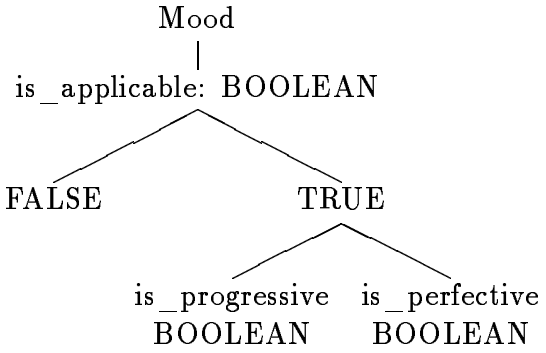


Les trois valeurs de Color indiquent respectivement des phrases à l'indicatif (*real*), au subjonctif ou au conditionnel (*unreal*), et à l'impératif (*order*). Ces valeurs n'ayant que peu de sens pour des phrases non-finies, il paraît raisonnable de n'activer le type Color que pour la valeur TRUE de *is_finite*.

Temps et Aspect

Le type Aspect peut être utilisé de façon optionnelle.[4] Dans le cas où l'aspect s'applique, deux possibilités se présentent, l'aspect perfectif et/ou progressif; chacun de ces deux champs peut prendre les valeurs booléennes TRUE et FALSE de façon indépendante :

(15)



[4]La valeur aspectuelle d'une forme infinitive telle que *manger*, par exemple, ne sera pas pertinente, si tant est qu'il soit possible de fixer une valeur aspectuelle (non-inhérente) pour ces formes. Cette approche est avant tout une question de commodité : la tâche sera simplifiée si on ne calcule pas la valeur aspectuelle dans certains cas, sans que cela affecte la génération.

Ce système relativement simple nous permettra cependant de traiter certaines facettes aspectuelles du système verbal français qui sont directement pertinentes pour le système.[5]

Pour traiter le système temporel, nous modifierons la représentation du temps adoptée à l'origine pour le système. Initialement, la représentation du temps dans les PSS suivait de façon stricte le système proposé par Reichenbach (1947). Pour mémoire, rappelons que ledit système distingue trois points: un point d'énonciation ou de parole (S), un point d'événement (E) et un point de référence (R). Deux relations sont également définies entre ces différents points: une relation de simultanéité (=) et une relation de précédence (<).[6] Les diverses combinaisons possibles dans un tel système ainsi que leurs correspondances avec les temps du français sont illustrées ci-dessous:

1. $E < R < S$ (Plus-que-parfait)
2. $E = R < S$ (Passé-simple; Imparfait).
3. $R < S < E$;$R < S = E$; $R < E < S$ (Conditionnel passé).
4. $E < S = R$ (Passé-composé).
5. $S = R = E$ (Présent).
6. $S = R < E$ (Futur proche).
7. $S < R < E$; $S = E < R$; $E < S < R$ (Futur antérieur).
8. $S < R = E$ (Futur).

Le système présenté présente en effet un certain nombre d'imperfections (cf Comrie 1981 pour une critique pertinente). Il existe par exemple une certaine redondance dans la définition du Conditionnel passé et du Futur antérieur, qui sont la réalisation de trois combinaisons temporelles différentes.

[5] Il est vraisemblable que le système devra être amélioré pour traiter des données plus complexes. Dorr & Gasteerland [1995] ont ainsi montré que l'aspect inhérent des items verbaux pouvait être pertinent en génération de surface, et Dorr & Broman Olsen [1996] que la télicité pouvait également jouer un rôle.

[6] Nous nous écarterons ici de la représentation traditionnelle (, et _) pour des raisons de lisibilité.

D'autre part, certaines conjugaisons telles que le Conditionnel présent ne sont pas exprimables sans faire appel aux définitions de Mood et de Color définies plus haut, en utilisant les valeurs Real et Unreal. Enfin, il est nécessaire de faire intervenir l'Aspect pour distinguer par exemple le Passé-simple de l'imparfait, qui ont, dans le système de Reichenbach, la même définition temporelle.

Une description plus élégante des temps verbaux du français devrait donc combiner le traitement de l'aspect, du mode et du temps pour donner de façon non redondante les formes conjuguées du français. L'arbre de la figure **??** définit une telle structure de données.[7]

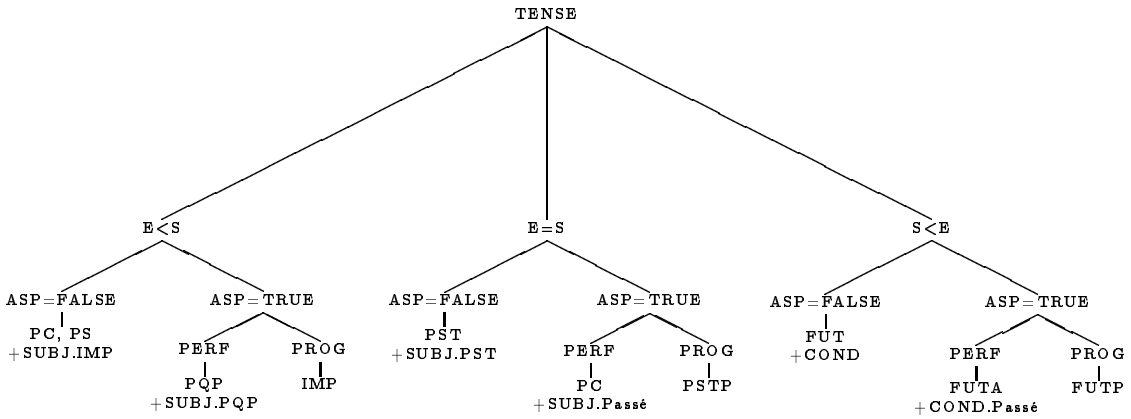


Figure 2.1: Structure temporelle

Ce système ne représente évidemment pas les phrases non-finies, infinitives et gérondives. Pour ces dernières, seul l'aspect est pertinent. Rappelons que le type Mood pose une variable *is_finite*; si celle-ci a la valeur FALSE, on obtient les infinitives et les gérondives. L'Aspect étant défini indépendamment de Tense, il est possible de faire jouer ce type pour distinguer ces constructions, comme ci-dessous :

[7]La liste des abbréviations est la suivante :
ASP : Aspect; PC : Passé Composé; PQP : Plus-que-parfait; IMP : Imparfait; PST : Présent; PSTP : Présent Progressif; FUT : Futur; FUTA : Futur Antérieur; FUTP : Futur Progressif; SUBJ : Subjonctif; COND : Conditionnel. Le symbole + indique que la valeur UNREAL est active.

- Mood : is_finite : FALSE; Aspect : is_applicable : FALSE ⇒ Forme infinitive.
- Mood : is_finite : FALSE; Aspect : is_applicable : TRUE; is_progressive : TRUE ⇒ Forme gérondive.
- Mood : is_finite : FALSE; Aspect : is_applicable : TRUE; is_perfective : TRUE ⇒ Forme infinitive aspectuelle (e.g., *avoir mangé*).

Les formes majeures du français sont ainsi aisément discriminées par le système.

Voix et Négation

Les champs Voix et Négation définissent simplement si la phrase est active ou non et négative ou non respectivement. Nous définissons donc deux variables booléennes *is_ active* et *is_ negated* :

- is_active : BOOLEAN
- is_negated : BOOLEAN

Si la variable *is_ active* a la valeur FALSE, les propriétés du passif (mouvement-A, etc., cf. la section 3.4 du chapitre 3) seront déclenchées; de même, la valeur TRUE pour la variable *is_ negated* déclenche l'insertion des marqueurs de négation (*ne* et *pas*).

Rôles thématiques, arguments, ajouts

Une autre caractéristique définitoire des CLS est la liste d'arguments du verbe, i.e. l'ensemble des éléments portant un des rôles thématiques assignés par le prédicat verbal. Il est donc nécessaire de spécifier la liste des θ-rôles assignés par le verbe ainsi que la liste des arguments portant ces rôles :

(16)a. V($(\theta_a$: 1)$(\theta_b$: 2)...$(\theta_z$: n))

b. Arguments: (Arg(1), Arg(2)...Arg(n))

Les deux listes sont ordonnées de façon à ce qu'une fonction simple fasse correspondre le premier θ-rôle de la liste thématique au premier Argument, et ainsi de suite.

Les arguments peuvent être de trois types :

1. DPStructure
2. CLS
3. SemanticLabel.

Les deux premiers cas d'arguments sont triviaux; il est bien connu que les groupes nominaux ainsi que les phrases peuvent fonctionner comme arguments d'un verbe, comme l'illustre l'exemple suivant :

(17) Marie veut que Jean vienne.

En (17), le DP *Marie* porte le rôle thématique Agent assigné par *vouloir* et la phrase *que Jean vienne* le rôle Theme. La DPStructure correspondant à *Marie* sera donc associée au θ-rôle Agent et la CLS associée à la complétive au θ-rôle Theme dans la CLS correspondant à la phrase principale.

Le troisième type d'argument concerne les exemples tels que (18) :

(18) J'ai mis le livre *sur la table*

Le PP *sur la table* est un argument du verbe, porteur d'un rôle thématique.[8] Or les PPs de ce type sont porteurs d'une fonction sémantique qui va au-delà de la simple relation thématique au verbe, ce qui nous a conduit à définir un type SemanticLabel caractérisant ces éléments. Il est donc nécessaire, d'une part de maintenir l'existence d'un type SemanticLabel dont les éléments ne fonctionnent normalement pas comme arguments d'un verbe, et d'autre part d'autoriser certains éléments de SemanticLabel à fonctionner comme arguments.[9] Pour simplifier la définition du type Argument, il serait

[8]Le test de constituant le plus simple montre que le PP est un argument du verbe, cf. l'impossibilité de l'omettre : **J'ai mis le livre.*

[9]Il semble que ce soient les éléments spatiaux qui aient la possiblité d'être des arguments de V, pour une classe de verbes limitée incluant notamment *mettre*.

possible de poser la relation *Argument=PSS*, en posant pour contrainte que seuls certains sous-types de SemanticLabel peuvent fonctionner comme arguments.

Enfin, il reste à inclure dans les CLS les éléments qui fonctionnent comme modifieurs, c'est à dire les éléments qui contribuent à l'information sémantique de la phrase mais qui ne sont pas porteurs de θ-rôles. Il est immédiat que ces éléments sont de type SemanticLabel; il ne reste donc qu'à inclure une liste (éventuellement vide) de ces éléments, comme décrit au paragraphe suivant.

Structure des CLS : Résumé.

(19)

```
CLS[

  Judgement :
    is_applied : BOOLEAN
                 Categorical : BOOLEAN

  Mood :
      is_finite : BOOLEAN
                  Color: (Real; Unreal; Order)
                           Tense: (E<S; E=S; S<E)


  UtteranceType : (declaration; question; exclamation)

  Verb : ...

  Thematic Roles : ...

  Aspect :
      is_applicable : BOOLEAN
              is_progressive : BOOLEAN
```

```
      is_perfective :  BOOLEAN

  Voice :
      is_active : BOOLEAN


  Negation :
      is_negated : BOOLEAN


  ArgumentList : LIST OF (DPStructure, CLS, SpatialSemanticLabel);

  ModifierList : LIST OF SemanticLabel;


]CLS
```

Cette structure décrit les éléments composant une représentation pseudo-sémantique des phrases. Les éventuelles modification des CLS sont laissées pour les recherches à venir. En l'état, les CLS nous permettront de générer les constructions majeures du français.

2.3.2 PSS II : DPStructure

La DPStructure spécifie les données suivantes :

- La structure de type opérateur-propriété d'un syntagme nominal (DP).
- Certains traits lexicaux associés au DP.
- Des indices associés au DP, qui comprennent un indice d'identification unique et un indice de coréférence.
- Une liste d'arguments.
- Une liste de modifieurs.

Une DPStructure sera donc représentée comme suit :

(20) ⟨ OPERATOR: ;PROPERTY: ;
REFERENTIAL-INDEX: (n, i);
Φ-FEATURES: Genre/personne/nombre;
LISTE D'ARGUMENTS
LISTE DE MODIFIEURS
⟩

Détaillons à présent chacun des composants des DPStructures.

Opérateur-Propriété

La représentation opérateur-propriété recouvre essentiellement la relation déterminant-nom, i.e. elle en est une représentation abstraite. Les noms ne seront pas transcrits en complexes de traits, pour des raisons pratiques, mais directement insérés sous le label Property (de même que les verbes apparaissent sous leur forme lexicale). Les déterminants par contre seront caractérisés de façon abstraite. La sémantique sous-tendant ces opérateurs est celle définie dans Barwise et Cooper [1981], reconsidérée et étendue par Keenan et Stavi [1986]. Pour l'essentiel, nous définissons un modèle extensionnel M doté d'un ensemble P de propriétés. Les noms dénotent des propriétés et les groupes nominaux des ensembles de propriétés. Les déterminants sont vus comme des fonctions de P dans P*, où P* est l'ensemble des sous-ensembles de P. En d'autres termes, les déterminants/opérateurs dénotent la façon d'associer une propriété (dénotation des noms) à un ensemble de propriétés (dénotation des groupes nominaux).

A titre d'illustration, l'opérateur ‖every‖ aurait la définition suivante :

(21) $\|\text{every}\|(A) =_{df} \{X \subseteq P \mid A \subseteq X\}$

où P est un ensemble non vide de propriétés et $A \subseteq P$. Cet opérateur est rebaptisé ‖every_global‖ dans le système, avec la définition donnée ci-dessus, de façon à le distinguer d'un opérateur quasi-équivalent, ‖every_particular‖. Ce dernier peut être défini comme suit :

(22) $\|\text{every\_particular}\|(A) =_{df} \{X \subseteq P \mid A \subseteq X \wedge [\forall\ Y \subseteq A: |Y|=1]\ \}$

Cette définition permet de s'assurer que l'ensemble final est un ensemble d'ensembles à cardinalité 1, ce qui recouvre le sens de *chaque*. [10]

Les opérateurs numéraux illustrent certains avantages de l'approche sémantique que nous adoptons. Un quantifieur généralisé du type *moins de trois personnes* n'est en effet simplement pas analysable en logique des prédicats avec des quantifieurs de premier ordre (cf Barwise et Cooper 1981); il est donc nécessaire de faire appel à une sémantique d'un autre ordre. A titre d'exemple, la définition des opérateurs $\|{=}n\|$ et $\|{\geq}n\|$ est la suivante :

(23)a. $\|{=}n\|(A) =_{df} \{ X \subseteq P \mid |A \cap X| = n \}$

b. $\|{\geq}n\|(A) =_{df} \{ X \subseteq P \mid |A \cap X| \geq n \}$

En d'autres termes, les opérateurs $\|{=}n\|$ et $\|{\geq}n\|$ sont des fonctions qui permettent d'aboutir à des ensembles de cardinalité n et supérieure ou égale à n.

Cette optique a pour avantage de permettre d'établir une correspondance quasi-univoque entre les fonctions définies sous le label Operator et les déterminants réels de la langue considérée (ici le français).[11]. Si l'on adoptait une analyse différente, en utilisant par exemple des traits du type [± indéfini] qui sont essentiellement des propriétés d'ensembles de fonctions, la correspondance avec les items lexicaux de la langue serait loin d'être univoque (cf. le cas de *certains*, *un*, etc. qui sont des indéfinis; cette caractéristique ne suffirait pas à les discriminer).[12]
Nous présenterons ici les fonctions associées aux principaux déterminants; les éléments Operator seront représentés entre barres verticales (à droite de la flèche se trouvent les réalisations de ces opérateurs en français) :

[10]Notons que la définition pourrait être réécrite en des termes ne faisant pas appel aux notions de logique des prédicats; la définition donnée ici semble plus simple et plus immédiatement accessible.

[11]Notons qu'à une fonction donnée peut correspondre une structure complexe du type *au moins 5*, ou 5 sera le déterminant syntaxique, modifié par le groupe adverbial *au moins*, lequel pourra se situer à droite ou à gauche du DP, voir plus bas.

[12]La définition d'opérateurs abstraits aura d'autre part une utilité évidente pour la traduction automatique: il est en effet souvent impossible d'obtenir une traduction satisfaisante en établissant une correspondance lexicale directe pour des expressions comme *more than five monkeys* ou *moins de cinq tables*.

1. $\|=n\| \Rightarrow$ (nombre) n.
2. $\|\geq n\| \Rightarrow$ au moins n.
3. $\|>n\| \Rightarrow$ plus de n.
4. $\|\leq n\| \Rightarrow$ au plus n.
5. $\|<n\| \Rightarrow$ moins de n.
6. $\|\simeq n\| \Rightarrow$ à peu près n.
7. $\|no\| \Rightarrow$ aucun, aucune.
8. $\|some\text{-}individual\| \Rightarrow$ un, une, des.
9. $\|\ some\_individual\_\leq\ \| \Rightarrow$ quelques.
10. $\|\ some\_individual\_\geq\ \| \Rightarrow$ plusieurs.
11. $\|the\| \Rightarrow$ le, la, les.
12. $\|every\_global\| \Rightarrow$ tout, toute.
13. $\|every\_particular\| \Rightarrow$ chaque.
14. $\|deixis\| \Rightarrow$ ce, cette, ces
15. $\|Poss\| \Rightarrow$ mon, ma , mes, etc. (classe des possessifs)
16. $\|WH\| \Rightarrow$ quel, quelle, quels, quelles.
17. $\|gen\| \Rightarrow$ le, la, les
18. $\|\Delta\| \Rightarrow \emptyset$

Quelques-uns de ces opérateurs méritent d'être détaillés.

En 12 et 13, les fonctions sont relativement semblables, bien qu'il soit possible de les distinguer: 12 est une fonction de l'ensemble des choses vers un ensemble d'individus (ayant une propriété commune), avec l'interprétation usuelle (cf B & C 1981, K & S 1987); la fonction en 13 est quasiment la même, hormis le fait qu'elle pointe vers un ensemble d'ensembles d'individus

ayant une propriété commune. En d'autres termes, 13 pointe vers chaque individu, qui constitue un ensemble, pour aboutir au final à un ensemble d'ensembles ayant les mêmes individus que pour l'ensemble créé par 12 (ce qui rejoint l'interprétation intuitive de quasi identité entre *tout* et *chaque*), bien que l'ensemble créé par 13 soit un ensemble d'ensembles d'individus.

Les fonctions en 9 et 10 requièrent également une explication détaillée. Elles sont avant tout une composition entre les fonctions $\|\geq n\|$, $\|\leq n\|$ et $\|some\_individual\|$. Ceci permet d'exprimer à notre sens le fait que *quelques individus écoutaient le discours* peut être paraphrasé en *il y avait au plus n individus* (notion d'existence réduite d'individus) , et *plusieurs individus écoutaient le discours* en *il y avait au moins n individus* (notion d'existence d'un nombre, supérieur à une certaine limite d'intérêt, d'individus).

Il est à signaler que les opérateurs $\|gen\|$ et $\|WH\|$, qui rendent compte respectivement des groupes nominaux génériques et interrogatifs, ne sont pas aisément définissables avec l'approche ensembliste et extensionnelle adoptée ici. Notons cependant qu'une analyse d'un autre type, intensionnelle, de ces déterminants serait également difficile à élaborer : la sémantique de la généricité et des déterminants interrogatifs est en effet un des problèmes majeurs toujours actuel en sémantique.

L'opérateur Δ est simplement la marque de non-spécification pour les opérateurs; en d'autres termes cet opérateur signale l'absence de tout opérateur, comme dans le cas des noms propres.[13]

Enfin, notons qu'il est aisé d'étendre la classe des opérateurs utilisée actuellement, en combinant ces opérateurs. Ainsi, la combinaison de l'opérateur $\|the\|$, qui est la fonction ayant pour domaine l'ensemble d'ensembles à cardinalité 1, peut se combiner à l'opérateur $\|=n\|$, donnant l'opérateur $\|the\ n\|$ qui permet de traiter les groupes nominaux du type *les trois chats*, par exemple. En combinant les fonctions avec des opérateurs booléens, il est également possible d'aboutir à de nouvelles fonctions. Ainsi, la combinaison $\|\geq n\| \wedge \|\leq m\|$ donne le déterminant:*entre n et m* (comme dans la phrase *entre 2 et 3000 manifestants, selon la police, ont manifesté hier après-midi*). D'autres opérateurs pourront également être définis directement, sans com-

[13]Il serait possible d'appliquer un autre opérateur aux noms propres, par exemple l'opérateur $\|every\_global\|$. Nous laissons de côté cette possibilité pour l'heure.

binaison d'opérateurs prédéfinis.

Traits lexicaux et indices

Voyons à présent les traits lexicaux devant être associés à DPStructure. Ce seront typiquement les traits ϕ, comprenant le genre, le nombre, la personne et éventuellement les distinctions de type [$\pm$ animé], etc. Bien évidemment, ce type d'information dépend essentiellement des caractéristiques des langues particulières considérées. En d'autres termes, étant donné qu'il existe une variation importante de langue à langue en ce qui concerne les traits ϕ, inclure ces derniers dans la pseudo-sémantique réduit la portée générale de ces structures. La question sera donc de n'inclure des traits ϕ que dans un petit nombre de cas pour lesquels il ne sera pas possible de faire autrement. Prenons comme hypothèse que l'on veut générer une phrase contenant des noms propres et des noms communs. Comme nous l'avons défendu plus haut, nous ferons le choix d'inclure directement ces items lexicaux (bien qu'il soit possible de procéder autrement). Or ces items ont chacun leurs traits ϕ associés dans le lexique: il ne sera donc pas nécessaire dans ces cas là d'inclure ces derniers au niveau pseudo-sémantique, à l'exception du trait de nombre, qui devra lui être mentionné.

Le cas des pronoms est cependant différent. Nous envisageons en effet de ne mentionner dans les PSS qu'une forme abstraite *pronoun* et non les items lexicaux proprement dits (*il, lui, etc.*).[14] Il sera dans ce cas nécessaire de mentionner les traits ϕ afin évidemment de pouvoir réaliser l'entité *pronoun* sous la forme d'un item lexical.

L'ensemble d'indices associé a une DPStructure comprend un indice d'identification unique et un indice de coréférence. Le premier permet de lier une structure DP à un des rôles thématiques attribués par la tête prédicative, et, plus généralement, servira d'identificateur du DP dans tout processus impliquant celui-ci. L'indice de coréférence joue un rôle actif dans le traitement des phénomènes de liage.

[14] Plus précisément, les pronoms seront la réalisation d'une DPStructure où l'opérateur et la propriété sont de type non-spécifié, Δ.

Arguments et modifieurs

Les deux listes inclues dans les DPStructures, arguments et modifieurs, caractérisent les éléments pouvant être associés à un DP. Les arguments sont en nombre très limités, et correspondent aux complétives du nom qui apparaissent avec certains noms abstraits, et aux compléments de noms relationnels, comme dans les exemples ci-dessous :

(24)a. L'idée que Jean viendra (me met mal à l'aise).

b. Le frère de Jean.

En (24a), le nom abstrait *idée* a pour propriété de pouvoir sélectionner un complément phrastique. La relation sémantique entre ces deux entités est loin d'être claire. Nous ne proposerons donc pour l'instant aucune analyse du phénomène, notant simplement que la phrase devrait être représentée comme un argument du nom. De même en (24b), la relation entre le nom *frère* et le PP *de Jean* est une relation de complémentation, comme pour tous les noms relationnels, i.e. qui imposent la présence d'un autre élément pour prendre son sens. Ce cas de figure impose là encore l'existence d'une liste d'arguments pour les DPStructures.

La liste de modifieurs permet d'intégrer les propositions relatives et tous les satellites prépositionnels du nom :

(25)a. La personne que Marie a vue

b. Marie, que nous connaissons tous

c. Le stylo de Marie

Les propositions relatives en (25a) et (25b) doivent être distinguées, la première étant une relative restrictive aux propriétés différentes des appositives, (25b). Nous ne détaillerons pas ici la représentation appropriée au niveau pseudo-sémantique pour les relatives, cette question réclamant une étude approfondie; notons simplement qu'il sera nécessaire de distinguer les deux cas de relativisation. Le cas de (25c), exprimant une relation de possession pourrait être caractérisé par le label POSS dont on donne deux illustration ci-dessous :

(26)a. (POSS | ⟨ OPERATOR: Δ;PROPERTY: Δ;
Φ-FEATURES: Masc, 1, sing;⟩)

b. (POSS | ⟨ OPERATOR: Δ ;PROPERTY: Marie ⟩)

Le premier exemple indique que dans la liste de modifieurs d'une DP-Structure (correspondant à *le stylo*, par exemple) il existe un élément exprimant le possesseur caractérisé par les traits (masculin, première personne du singulier). La réalisation de ces spécifications serait dans ce cas le DP *mon stylo* (pronominalisation possessive). En (26b), la réalisation serait *le stylo de Marie*.[15]

2.3.3 PSS III : SemanticLabel

Les élément de type SemanticLabel servent intuitivement à caractériser les objets qui contiennent de l'information sémantique mais ne sont pas liés à un rôle thématique (cf. cependant les cas exceptionnels décrits plus haut.). En français, ce sont typiquement les groupes prépositionnels et les phrases circonstancielles :

(27)a. Les étudiants travaillent **derrière la vitre**.

b. Il faut avancer **avant que les ennuis ne commencent**.

Dans ces deux exemples, le groupe prépositionnel et la circonstancielle ajoutent de l'information sémantique, respectivement spatiale et temporelle. Comme nous le verrons par la suite, la réalisation de SemanticLabel peut également être non prépositionnelle ou phrastique. Les caractéristiques définitoire de SemanticLabel sont les suivantes :

Définition. Un type SemanticLabel comprend :

- Une **valeur** définissant les propriétés sémantiques.
- Un **type** pouvant être DPStructure ou CLS.

[15] Notons que cette approche, qui paraît souhaitable, impliquerait d'éliminer la classe des déterminants possessifs de la classe Operator.

La valeur désigne la propriété sémantique, qui peut être réalisée par une préposition (cf. *derrière* en (27a)), un complémenteur (*avant que* en (27b)), ou un adverbe (e.g., la valeur UNDER, que l'on verra plus loin, peut avoir la réalisation adverbiale *dessous*).

Le type inclu dans le type SemanticLabel sert à indiquer la structure associée à la valeur, qui peut-être une DPStructure ou une CLS (permettant de générer un groupe prépositionnel avec le DP associé ou une circonstancielle). Ainsi, en anticipant sur la suite de la présentation, la valeur FOR peut être associé à une DPStructure ou à une CLS, ce qu'exprime le SemanticLabel en (27a), donnant par exemple les réalisations en (27b) et (27c) :

(28)a. SemanticLabel(FOR | DPStructure/CLS)

b. pour mes camarades

c. pour que Marie dorme

Nous distinguerons les trois sous-types de SemanticLabel suivants :

- GeneralLabel
- SpatialLabel
- TemporalLabel

Cette classification en termes de labels spatiaux, temporels et généraux pourrait bien évidemment être raffinée, mais elle nous servira en l'état à classifier et traiter les cas principaux du français.

GeneralLabel

Les labels généraux ont pour l'instant les définitions suivantes :[16]

1. (WITH | DPStructure)

(29) Je viendrai **avec Marie**

[16]Nous donnons pour chaque label sa valeur, la (les) structure(s) associable(s) et des exemples de réalisation.

2. (WITHOUT | DPStructure/CLS)

(30)a. Je dormirai **sans Marie**

b. Je dormirai **sans que mon chef ne le sache**

3. (AGAINST | DPStructure)

(31) Jean travaille **contre le capitalisme**

4. (FOR | DPStructure/CLS)

(32)a. Jean lutte **pour l'anarchie**

b. Je donnerai mon travail à la machine **pour qu'il soit fait plus rapidement**

5. (MODU | DPStructure/CLS)

(33)a. Je travaille **selon le temps**

b. Je ferai mon travail **selon qu'on me paye ou non**[17]

6. (MODU_OPP | DPStructure/CLS)

(34)a. Je travaille **malgré le beau temps**

b. Il est venu **bien que la pluie soit tombée abondamment**

7. (POSS | DPStructure)[18]

8. (PROGACT | CLS)

(35) Je travaillerai **en chantant**

9. (OPP | CLS)

[17]Ce label donne lieu à des tournures *phrastiques* particulières en ce sens que la présence de la locution *ou non* en fin de phrase est quasi obligatoire.

[18]Cf. la section précédente.

(36) Il dormait **alors que le soleil brillait au dehors**.

10. (LINK | CLS)

    (37) Il y aura des malheureux **tant que le système sera ce qu'il est**.

11. (HYP | CLS)

    (38) Il viendra **s(i)'il n'y a pas de travail à faire**.

12. (CAUSE1 | CLS)

    (39) Il a travaillé **parce qu'il n'avait pas le choix**.

13. (CAUSE2 | CLS)

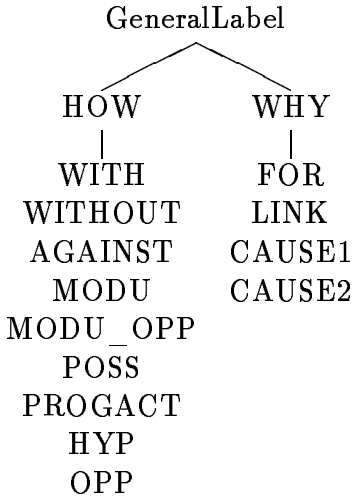
    (40) **Puisque tu ne veux pas travailler**, je m'en vais.

Le label 1 définit une relation d'ajout d'élément au procès, lequel élément peut-être abstrait (e.g. appréciation subjective, *avec plaisir*) ou concret (*avec la voisine*). Le label en 2 définit *grosso modo* l'inverse. 3 indique l'opposition. Le label en 4 définit l'inverse du précédent. Les labels en 5 et 6 définissent une appréciation de l'événement (au sens large) décrit dans la SPS modifiée: MODU relativise le cadre de l'événement, tandis que MODU_OPP indique le type d'obstacle franchi par l'événement. Le label 8 correspond aux structures gérondives. 9 caractérise les phrases marquant l'opposition à l'énoncé principal. Le label 10 définit les phrases établissant un lien de simultanéité *et* de causalité avec l'énoncé principal. 11 vaut pour les phrases hypothétiques. Le label 12 définit les phrases donnant la cause de l'événement décrit dans la principale. Enfin, 13 est le cas de figure inverse du précédent, la principale explicite de quoi l'événement décrit dans la circonstancielle est la cause.

Il reste à présent à considérer le cas des adverbiaux interrogatifs, du type *comment*, *pourquoi*, *quand* et *où*. Les deux derniers caractérisent respectivement des fonctions temporelles et spatiales, que nous verrons par la suite. Tous quatre correspondent cependant au questionnement sur les grands types de classification des "événements" au sens large du terme. Les

labels généraux définis ci-dessus correspondent en fait à des sous-types répondant à la question *Pourquoi?* ou à la question *Comment?*. Il est donc nécessaire de définir deux types HOW et WHY, qui seront les sous-types principaux du type GeneralLabel (lui-même sous-type de SemanticLabel). L'organisation est ainsi la suivante :

(41)



Ceci est bien entendu une approximation et ne saurait être pris comme une catégorisation définitive des concepts sémantico-cognitifs. Pour prendre une exemple concret, le label OPP peut signifier bien autre chose que la "réponse" à la question *comment?*. Nous ne chercherons pas à caractériser plus finement les éléments ici définis. Un résultat plus directement important est la possiblité de réaliser les sous-types HOW et WHY comme des adverbes. Ainsi, HOW peut se réaliser comme l'adverbe interrogatif *comment* ou comme les adverbes de manière *facilement*, *bruyamment*, etc.; WHY par contre ne se réalise que sous la forme interrogative *pourquoi*. Ces deux types ont donc pour caractéristique de n'avoir pour réalisation de leur valeur que des formes adverbiales lorsqu'ils ne "branchent" pas sur un sous-type.

SpatialLabel

Comme pour les labels généraux, les labels spatiaux définissent un type primordial de label, le label WHERE. Celui-ci peut-être réalisé sous la forme de

l'adverbe *où*, ou comme un des (sous)-labels spatiaux dont la liste provisoire est donnée ci-dessous. Nous modifierons cette liste, qui compte les valeurs standards de la spatialisation, en cours d'examen des prépositions.[19]

1. UNDER: sous/dessous
2. ON: sur/dessus
3. AROUND: autour
4. BEHIND: derrière
5. IN FRONT: devant
6. IN: dans/chez/à
7. OUT: hors/dehors
8. AMONG: parmi
9. AGAINST: contre
10. ORIGIN: de
11. DIRECTION: à/vers
12. BETWEEN: entre

Certains des labels ci-dessus méritent une explication détaillée. Considérons tout d'abord le cas de UNDER et ON, qui ont pour correspondants une préposition (respectivement *sous* et *sur*) et un adverbe (resp. *dessous* et *dessus*). Ceci étant une particularité du français, la distinction ne se fera pas en termes de deux labels différents.[20]

Les labels AGAINST et ORIGIN correspondent respectivement à des emplois du type *La tête contre le mur* et *De Paris à Londres*, i.e. le premier

[19] Aucun label spatial ne peut en effet comprendre une CLS. Seules des formes adverbiales ou prépositionnelles sont produites pour réaliser la valeur de ces labels. Pour simplifier la présentation, nous donnons la valeur du label et sa réalisation prépositionnelle ou adverbiale.

[20] En anglais, par exemple, les adverbes ne sont pas tolérés pour ON et UNDER, cf **?? Put it under!* par rapport à *Mets-le dessous!*.

désigne une relation spatiale de contact entre éléments, le deuxième marque le point d'origine d'un déplacement essentiellement. Ceci n'implique évidemment pas que ces prépositions n'aient pas d'autres emplois, voir plus bas.

Les deux réalisations possibles de DIRECTION, *à* et *vers*, toutes deux des prépositions, imposent une analyse plus fine. Il se pourrait que ces deux réalisations soient simplement des variantes non distinguables, comme semblent le montrer des phrases comme *Je vais vers le/au Sud.* Cependant, il existe bien des cas où l'une semble préférée, comme dans *Les policiers coururent vers/??à moi.* D'un point de vue intuitif, *vers* à un emploi plus centré sur la direction elle-même, tandis que *à* dispose d'un emploi plus large : dans la phrase *Il est allé au cinéma*, l'interêt peut être centré sur la direction, étant dans ce cas quasi identique à *vers*, ou indiquer que l'élément déplacé se trouve dans un nouveau lieu, cette deuxième interprétation étant à notre sens largement préférée. Il découle de ce fait que l'emploi de *à* paraît déplacé dans une phrase où est décrit un déplacement vers un élément non susceptible de contenir l'élément déplacé. En d'autres termes, le *à* directionnel s'emploie préférentiellement pour indiquer un changement de lieu d'un individu, plutôt que sa direction: dans la phrase *Il est rentré à la maison*, la direction importe relativement peu, l'information principale étant que l'individu en question se trouve dans un lieu différent, après s'être déplacé. De ce fait, la phrase *??Il est allé à la voisine* est étrange, l'élément *voisine* n'étant pas a priori susceptible de désigner un lieu où se trouvera l'individu *il* après s'être déplacé.[21] La préposition *vers* n'ayant que l'emploi directionnel, la nature du point de destination, contenant potentiel ou non, importe peu (cf. *Il est allé vers elle/la gare*). Nous distinguerons donc les deux labels suivants:

1. DIRECTION_CENTERED: vers
2. DIRECTION_IN: à

Pour éviter de générer les phrases du type **Il s'est dirigé à elle*, il serait nécessaire de tenir compte de la nature contenant/non contenant de l'élément Property lié au label DIRECTION, de façon à toujours sélectionner

[21]Ceci cesse évidemment d'être vrai dès que le verbe de déplacement dépasse son strict sens de déplacement. Ainsi, dans la phrase *Ils sont venus à moi comme des agneaux apeurés*, le sens métaphorique de contenant/refuge du terme *moi* autorise l'emploi de *à*.

vers lorsque la Property n'est pas un contenant potentiel pour un élément. Nous laisserons de côté ce type de raffinement pour l'instant.

Le label IN donne également lieu à plusieurs réalisations potentielles *dans*, *chez* et *à*. Le premier cas correspondrait à des phrases telles que *Je suis dans la salle de bain*, le deuxième à *Je suis chez la voisine*, le troisième à une phrase du type *Je suis à la maison*. Comme on le voit sur la base de ces exemples, il n'est pas possible de discriminer les réalisations de IN sur la simple base de la sémantique du verbe. En première analyse, on pourrait avancer que la propriété pertinente pour la distinction est le caractère plus ou moins animé de l'élément Property lié au modifieur spatial: si cet élément est [-animé], la réalisation de IN sera *dans* ou *à*, dans le cas inverse la réalisation sera *chez*. [22] Notons que cela n'interdit pas de générer *Le microbe était dans son corps*, car il semble que l'on ne considère pas *corps* comme un élément animé, bien qu'il le soit indubitablement, seul l'individu doté du corps étant considéré comme animé. D'autre part, pour pouvoir parler de choses qui sont dans des individus mentionnés comme tels, il est remarquable que le français utilise une autre forme, à savoir *en*, comme dans *La stupidité est en/*dans toi*; ceci indique apparemment la réticence à employer *dans* pour indiquer la présence d'un élément dans un être animé. Pour distinguer *en* de *chez*, il est là nécessaire de percevoir la différence entre la relation IN établie réellement sur l'individu [+animé] (i.e. *en* établit un lien d'inclusion d'un élément dans l'individu animé proprement dit) et la relation IN lorsque l'individu désigne, par raccourci, le lieu-contenant proprement dit (i.e. *chez la voisine* est un raccourci pour *dans la maison de la voisine*).Ce type de connaissance-inférence pragmatique dépassant les limites du système de représentation des connaissances envisagé ici pour la génération, nous ne distinguerons pas les deux formes *en* et *chez*.

La distinction est encore difficile en ce qui concerne *dans* et *à*. Il semble cependant que les questions de focalisation sur le lieu soient pertinentes ici, tout comme la question de la focalisation sur la direction était pertinente pour distinguer *vers* et *à*: *dans* focalise clairement l'attention sur le lieu

[22] Notons que nous nous interdisons ainsi de générer des phrases du type *J'ai passé l'après-midi dans la voisine* qui peuvent être concevables avec un contexte approprié. Seules des considérations pratiques de génération (et non des considérations de bienséance) nous conduisent à ce choix.

tandis que *à* a une fois de plus un emploi plus "lâche". Une analyse plus précise de la distinction nous emmenerait cependant trop loin; nous nous contenterons donc définir les labels suivants:

1. IN_CENTERED: *dans* si PROPERTY=[-animé]; *chez*/*en* si PROPERTY=[+animé]
2. IN_WEAK: *à*

Notons pour clore cette section que nous n'avons pas eu recours à des distinctions du type [± mouvement], etc., car même les prépositions qui ne relèvent apparemment pas de la désignation d'un mouvement (contrairement à *vers*, par exemple), telles que *sur* ou *sous*, peuvent être aisément employées avec des verbes de mouvement, cf *Il marchait sur le toit.* D'une manière générale, les labels définis ici permettent une couverture large des divers emplois spatiaux des prépositions, tout en définissant une correspondance quasi univoque avec les items lexicaux de la langue. Le système contera donc les types suivants pour SpatialLabel :

1. UNDER: sous/dessous
2. ON: sur/dessus
3. AROUND: autour
4. BEHIND: derrière
5. IN FRONT: devant
6. IN:
   (a) IN_CENTERED: *dans* si PROPERTY=[-animé]; *chez*/*en* si PROPERTY=[+animé]
   (b) IN_WEAK: *à*
7. OUT: hors/dehors
8. AMONG: parmi
9. AGAINST: contre

10. ORIGIN: de
11. DIRECTION:
    (a) DIRECTION_CENTERED: vers
    (b) DIRECTION_IN: à
12. BETWEEN: entre

TemporalLabel

Le type TemporalLabel, comme les deux types vus précédemment, définit un premier sous-type essentiel, le label WHEN, qui peut être réalisé comme l'adverbe *quand*, ou comme un des sous-types que l'on définira dans ce qui suit.

Les sous-labels de WHEN seront définis suivant une organisation autour de repères temporels. A titre d'illustration, la préposition *avant* place l'évènement avant un repère temporel défini par le complément de la préposition: ainsi, dans la phrase *Il viendra avant la fin du repas*, l'évènement de la venue de *il* se place avant le repère temporel de la fin du repas. Pour rendre compte d'autres emplois, il est nécessaire d'introduire également des notions telles que la durée, désignant le fait pour un point temporel d'être spécifiquement considéré en tant que durée (cf. la phrase *Il a écrit ses rapports en 8 ans*). Une première analyse nous donne la classification suivante, par rapport au trait *duration* :

1. Trait [-duration] : en, dans, avant, après, hier, demain, etc.
2. Trait : [+duration] durant, pendant, en, dans

Analysons tout d'abord le cas des éléments [+duration]. Un premier critère peut nous permettre de distinguer la préposition *en* des autres. En effet, une certaine notion de vitesse d'accomplissement d'évènement sous-tend cette préposition, la vitesse pouvant être lente ou rapide. Ainsi, dans la phrase *Il a écrit ses rapports en 8 ans*, l'ensemble *en 8 ans* peut impliquer une certaine lenteur dans l'exécution (i.e. l'auteur n'est vraiment pas rapide), ou inversement une certaine rapidité (i.e. étant donné l'ampleur de

la tâche, c'est un exploit), suivant les informations contextuelles. Cette notion de vitesse n'est présente ni avec *durant*, ni avec *pendant*. Le cas de *dans* est plus problématique quant à ce problème particulier. On peut en effet sous entendre la vitesse d'exécution de la tâche dans la phrase *J'ai fini mes rapports dans la nuit*. Il semble cependant que cette interprétation ne soit pas définitoire pour *dans*, contrairement à *en* (cf. l'impossiblité d'avoir **Je les ai écrites dans 5 heures*).

Le premier résultat sûr concernant les éléments [+duration] est donc le suivant :

(42) [+duration]/[+speed]= en (/dans)

Le cas de *durant* et *pendant* pose problème. Il est en effet à notre sens quasiment impossible de distinguer ces deux items dans l'optique abordée ici. Tous deux définissent une durée temporelle et aucune idée de vitesse de réalisation n'est présente. Il semble que seul le niveau de langue varie, *durant* étant plutôt lié à un style littéraire et *pendant* à un parler courant. Nous laisserons donc de côté pour l'instant la réalisation *durant*, en gardant cependant à l'esprit ce cas de figure lorsque le générateur sera doté d'une option de niveau de langue.

(43) [+duration]/[-speed]= pendant

L'utilisation de *pendant* est illustrée ci-dessous:

(44)a. Il a lu pendant la/des/plusieurs/au moins 10/ nuit(s)

b. Le diable vient pendant mon sommeil/mes vacances

Considérons à présent le cas des éléments [-duration]. L'idée générale est que la modification temporelle s'organise dans ce cas autour d'un certains nombre de repères temporels. Nous définirons les repères suivants :

1. E_P : point d'évènement de la CLS principale.
2. E_{Sem} : point d'évènement défini par le label.[23]

[23]Le terme "évènement" est pris au sens large. Ainsi, *la pluie* pourra définir un point d'évènement.

3. S : moment d'énonciation.

4. D_{ABS} : date temporelle définie par le label.

5. D_{GEN} : date générique.

6. $INTERVAL_{E\_P-Sem}$: intervalle défini entre E_P et Sem.

Ces repères ne nous intéressent pas dans l'absolu mais uniquement dans leur organisation en système. Les interactions suivantes entre les points ci-dessus seront les labels intégrés au générateur :

1. ($E_P < E_{Sem}$ | DPStructure/CLS)

   (45)a. Je pars **avant la pluie**.

   b. Je pars **avant que mes amis arrivent**.

2. ($E_P > E_{Sem}$ | DPStructure/CLS)

   (46)a. Je pars **après la pluie**.

   b. Je pars **après avoir mangé**.

3. ($E_P = E_{Sem}$ | CLS)[24]

   (47) Un travail me déprime **quand j'ai à le faire**.

4. ($E_P <$ S | DPStructure/$\emptyset$)[25]

   (48) J'ai travaillé **hier/avant-hier/il y a dix ans**.

5. ($E_P >$ S | DPStructure/$\emptyset$)

   (49) Je travaillerai **demain/après-demain/dans dix ans**.

6. ($E_P =$ S | $\emptyset$)

[24]Nous laissons de côté les expressions du type *en même temps que les enfants*, dont le statut structural n'est pas clair.

[25]La désignation $\emptyset$ indique que c'est le label qui est réalisé, sous forme adverbiale.

(50) On arrête de travailler **maintenant**.

7. (E_P = D_{ABS} | DPStructure)

   (51) Je viendrai **lundi/en 1998**

8. (E_P = D_{GEN} | DPStructure)

   (52) Je dors **chaque lundi matin/le dimanche**

9. (INTERVAL$_{E_P<E_Sem}$ | DPStructure/CLS)

   (53) Je ne dors plus **depuis que l'on m'a privé de travail/depuis les bombardements.**

10. (INTERVAL$_{E_P>E_Sem}$ | DPStructure/$\emptyset$)

    (54) Ils sont partis **pour 10 jours**

Pour conclure cette section, notons deux points importants. D'une part les labels définis ici ne sauraient rendre compte de la représentation temporelle dans son ensemble. D'autre part, la partie la plus délicate dans le traitement de ces labels concerne leur réalisation effective dans la langue. Le choix d'une préposition adéquate dépend par exemple fréquemment du nom temporel choisi, comme l'illustrent les cas de figure suivants :

- Soit le label : (E_P = D_{ABS} | DPStructure),
- La réalisation du label est du type [α time], avec :
  - SI time=année (1997,etc.) $\vee$ mois(janvier, etc.) $\Rightarrow$ α=en

    (55) En janvier 1998
  - SI time=date incluant nombre du jour du mois (e.g. 3 janvier) $\vee$ nom temporel modifié (e.g. premier lundi du mois, mois prochain) $\Rightarrow$ α=le

    (56) Le 3 janvier 1998

- SI time=jour de la semaine $\Rightarrow \alpha=\emptyset$

  (57) Mardi

- SI time=jour de la semaine $\wedge$ indéfini $\Rightarrow \alpha$=un

  (58) Un vendredi

- SI time=heure et dérivés (demi-heure etc.) $\Rightarrow \alpha$=à

  (59) A quatre heures

Comme on le voit, la génération automatique des formes temporelles appropriées réclamera un calcul relativement complexe sur les PSS, et un examen complet de tous les éléments composant l'expression temporelle pour pouvoir générer des expressions temporelles bien formées.

2.4 Conclusion

La représentation des connaissances adoptée pour le système GBGen et présentée dans ce chapitre permet de satisfaire certaines contraintes que nous avions imposées à ce point d'entrée de la génération.

Tout d'abord, il était nécessaire de s'éloigner, dès que possible, des propriétés particulières des langues ainsi que des traits purement syntaxiques. Ceci a été accompli dans le traitement des déterminants, du temps, de l'aspect, des rôles thématiques, des prépositions et des circonstancielles. Dans tous ces cas, nous avons défini des catégories plus générales, valables pour un certain nombre de langues, et non uniquement pour le français, et permettant en même temps d'engendrer des réalisations particulières de façon directe.

D'autre part, le système est tout à fait gérable du point de vue computationnel. Suivant l'idée de base de la pseudo-sémantique énoncée dans Clark [1993], nous avons en effet renoncé à la décomposition lexicale des items de classe ouverte (verbes, noms, etc.) et au calcul des inférences sémantiques complexes.

Enfin, la définition des structures pseudo-sémantiques peut aisément être enrichie afin de traiter de nouveaux phénomènes, notamment les adjectifs et les structures partitives, entre autres.

dans le chapitre suivant, nous étudierons la production des structures syntaxiques sur la base de ces informations pseudo-sémantiques, ainsi que les diverses opérations s'appliquant sur les arbres syntaxiques.

Chapter 3

Génération automatique

> "To err is human, but to really foul things up
> requiers a computer".
> *Murphy's Laws of Technology.*

Sur la base des informations détaillées au chapitre précédent, le système doit produire des structures syntaxiques bien-formées. Comme nous l'indiquions dans l'introduction, le système est conforme au modèle-T de la théorie des principes et paramètres (cf. Chomsky 1981), et produit une structure profonde comme première représentation syntaxique. Nous discuterons à la section 3.5 du bien fondé de ce choix. Dans l'immédiat, nous décrirons le fonctionnement du système de projection des structures syntaxiques.

3.1 De la pseudo-sémantique à la syntaxe

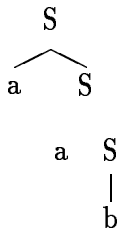
Depuis les premiers travaux sur les grammaires formelles, l'utilisation de règles de réécriture pour la production de structures syntaxiques est le formalisme dominant. Ainsi, la grammaire en (60a) dotée des règles en (60b) génère le langage (60c) dont une des phrases pourrait être (60d).

(60)a. G=(V_A, V_T, R, S)

b. R: S -> aS
S -> b

c. L={a*b}

d.

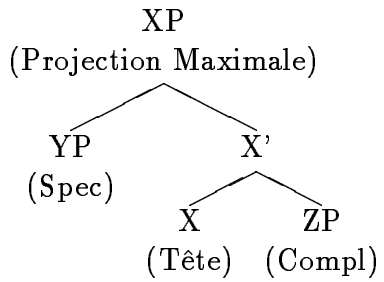


Pour créer la structure en (60d), on applique deux fois la première règle (S -> aS) et une fois la deuxième (S -> b). Ces règles de réécriture constituent un formalisme efficace pour créer des structures syntaxiques et sont du reste toujours couramment utilisées dans de nombreuses théories de la grammaire (cf. les grammaires d'unification, par exemple). Depuis le début des années 80, la conception de la grammaire a été modifiée dans le cadre chomskyen, par l'introduction de l'idée de *Projection* (cf. Stowell 1981), qui pose que les structures sont formées à partir des propriétés des items lexicaux. La grammaire GB a pour composant central le Principe de Projection, que l'on peut formuler informellement comme suit :

Principe de Projection. Les propriétés lexicales doivent être représentées à tous les niveaux de représentation syntaxique.

L'idée de ce principe est avant tout d'établir la correspondance entre propriétés lexicales et représentations syntaxiques. Les n propriétés thématiques d'un verbe, par exemple, imposeront la présence de n arguments en syntaxe. Ce principe a de multiples conséquences (notamment la nécessité de recourir à des catégories vides), mais nous nous bornerons ici à étudier la création des structures syntaxiques. Comme l'indique le nom du principe, les structures syntaxiques sont censément *projetées* à partir des propriétés lexicales des items, pour aboutir à une structure conforme au schéma X-barre donné ci-dessous :

(61) Schéma X-barre :



La tête X est le noyau de cette représentation, et projette à deux niveaux : i) le niveau X' (ou X-barre) regroupant la tête et son ou ses compléments, et ii) le niveau maximal XP comprenant un Spécifieur et le niveau X'.

Le schéma X-barre présente l'avantage d'uniformiser les structures syntaxiques, assurant une régularité dans le traitement des structures. Au niveau de la *création* des structures syntaxiques l'idée de projection n'est cependant qu'une question de terminologie, du moins dans les versions standards de la théorie GB. Le schéma X-barre est en effet produit par des règles de réécriture comme les suivantes :

(62) XP $\rightarrow$ YP X'
X' $\rightarrow$ X ZP

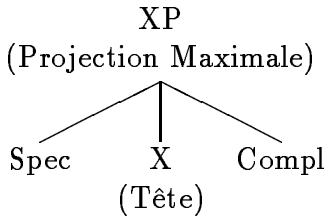
Le fait d'avoir ces règles de réécriture contredit la caractérisation intuitive du terme de *Projection*, qui suppose une projection des structures à partir des items lexicaux ("bottom-up") alors que la création des structures par application des règles de réécriture est descendante ("Top-down"). Contrairement à l'idée répandue que la théorie X-barre définit un nouveau type de grammaire, ou un nouveau type de création de structures, il ne s'agit là que d'une grammaire contexte-libre classique (cf. Pullum 1986 pour une mise au point sur cette question).

Il est à noter que les techniques de création ascendante de structures syntaxiques en analyse sont bien connues, et sont même plus nombreuses que les techniques descendantes (cf. notamment la classe LR(k) parmi les langages algébriques). Ces techniques/formalismes sont cependant intimement liés à la notion d'analyse et notamment à l'utilisation d'automates à pile. D'un point de vue formel, la plupart de ces techniques supposent l'existence de

règles de réécriture, que l'on traite en tentant de réduire la partie droite des règles à la partie gauche.[1] Il n'existe pas, à notre connaissance, de description formelle des mécanismes de projection ascendante des structures syntaxiques indépendamment des outils formels d'analyse.[2] Nous n'entamerons pas ici une telle formalisation, une définition complète et satisfaisante d'un point de vue formel des mécanismes de projection ascendants étant un travail de grande ampleur qui dépasserait largement le cadre de ce mémoire; il est du reste non trivial de définir formellement de tels mécanismes de projection ascendants.[3] Nous nous contenterons donc ici d'établir une correspondance entre les propriétés des éléments des PSS avec les représentations syntaxiques, en accord avec l'idée *intuitive* du Principe de Projection.

Par commodité d'implémentation, le schéma X-barre que nous adoptons est le suivant :

(63)



Spec et Compl sont des listes, éventuellement vides, de projections maximales.[4] Ces listes, suivant l'idée du Principe de Projection, doivent être remplies par des éléments satisfaisant des propriétés de la tête X. Par commodité, nous continuerons à utiliser l'expression "projeter" pour décrire le

[1] Les approches *déplacement/réduction* illustrent cette technique.

[2] Cf, cependant Tsoulas [1997] qui propose une formalisation du programme minimaliste basée sur la théorie des ensembles. La création des structures n'est toutefois pas celle que nous envisageons ici.

[3] Précisons qu'il est toujours possible d'associer aléatoirement des éléments lexicaux, en utilisant l'opération simple de concaténation, comme le propose Chomsky [1995]. Les difficultés se présentent lorsque l'on veut que le mécanisme soit déterministe, projettant les structures d'après les propriétés des éléments lexicaux.

[4] La représentation "plate" du schéma X-barre n'implique pas que les relations de c-commande soient informulables, cf. Merlo [1993].

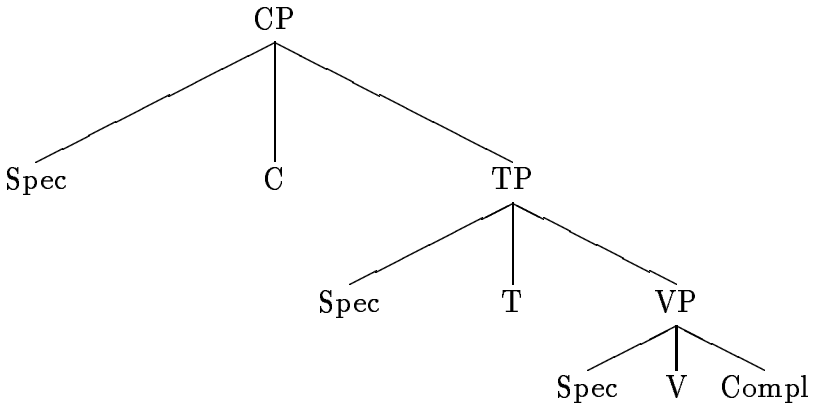
passage d'une tête X prise dans le lexique à une structure de type XP.

Nous établirons les correspondances principales suivantes entre pseudo-sémantiques et syntaxe :

- V $\Leftrightarrow$ V
- PROPERTY $\Leftrightarrow$ N
- OPERATOR $\Leftrightarrow$ D [5]
- Valeur de SemanticLabel $\Leftrightarrow$ P ou C
- UtteranceType $\Leftrightarrow$ C
- Tense $\Leftrightarrow$ T

Pour chacune de ces correspondances, le système projette une structure XP. En laissant de côté pour l'instant le cas des groupes nominaux et prépositionnels, la structure d'une phrase sera schématiquement la suivante :

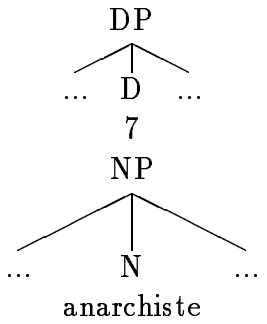
(64)



Voyons à présent le détail de la satisfaction des propriétés des têtes. Supposons que le système présente à l'entrée une PSS du type DPStructure, avec OPERATOR=‖=7‖ et PROPERTY=‖anarchiste‖. Le processus de projection créera les deux sous-arbres suivants :

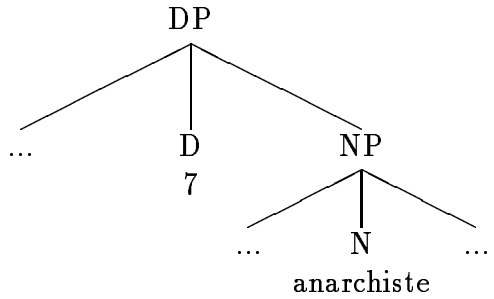
[5] Ceci est une simplification des correspondances. Comme on l'a vu au chapitre 2, certains opérateurs peuvent avoir pour réalisation une structure plus complexe du type *plus de trois*.

(65)



Il est maintenant nécessaire de considérer les propriétés des composants de ces arbres. Ainsi, D a pour propriété lexicale de sous-catégoriser un NP. Cette propriété déclenche l'insertion du NP en Compl du DP, donnant :

(66)



Ce type d'opération d'insertion d'un sous-arbre dans un autre est assez semblable dans l'esprit aux TAGs (Joshi 1987). Sur la base de l'exemple précédent, nous pouvons décrire une procédure générale de combinaison des arbres :

(67) Soient les sous arbres XP et YP, si YP satisfait une propriété de la tête X, insérer YP dans une des listes (Spec, Compl) de XP.

Les propriétés à satisfaire sont :

- Les propriétés thématiques (pour les verbes),
- Les propriétés de sous-catégorisation/sélection,

- Les relations pseudo-sémantiques (e.g., les relations de modification, i.e. le fait pour un élément d'apparaître dans la liste des modifieurs d'un élément dans une PSS donnée, les relations Opérateur-propriété, etc.)

Le placement des projections dans les listes Spec ou Compl dépend avant tout du type de propriété satisfait par les projections. Ainsi, si un groupe nominal satisfait à la fois une propriété thématique non agentive du verbe, il sera généré en Compl de VP (cf. Larson 1988). Le cas des ajouts/modifieurs étant plus variable, nous le laisserons de côté.

La procédure générale peut donc se résumer comme suit :

(68) Soit une PSS :

- Associer les éléments de la PSS à leurs correspondants lexicaux catégoriels,
- Projeter chaque item au niveau XP (donne l'ensemble $\mathcal{M}$ des projections),
- Pour toutes les paires de sous-arbres XP et YP appartenant à $\mathcal{M}$, si YP satisfait une propriété de la tête X, insérer YP dans une des listes (Spec, Compl) de XP.

Pour prendre un exemple concret, considérons la PSS suivante :

(69)

```
PSS(P) [
   Mood           : finite
   -Color         : real
   -Tense         : S = E
   Aspect         : FALSE
   Voice          : active
   Negation       : not negated
   Utterance type : declaration
```

```
   Verb           : regarder
   Thematic roles : (agent theme)
   Arguments      :
   Arg(1) {
      <
         Property         : Jean
         Operator         : delta
         Number           : singular
      >
   } Arg(1)
   Arg(2) {
      <
         Property         : télé
         Operator         : 'THE' determiner
         Number           : singular
      >
   } Arg(2)
] PSS(P)
```

Les deux premières étapes de la procédure nous donnent l'ensemble de projections suivant :

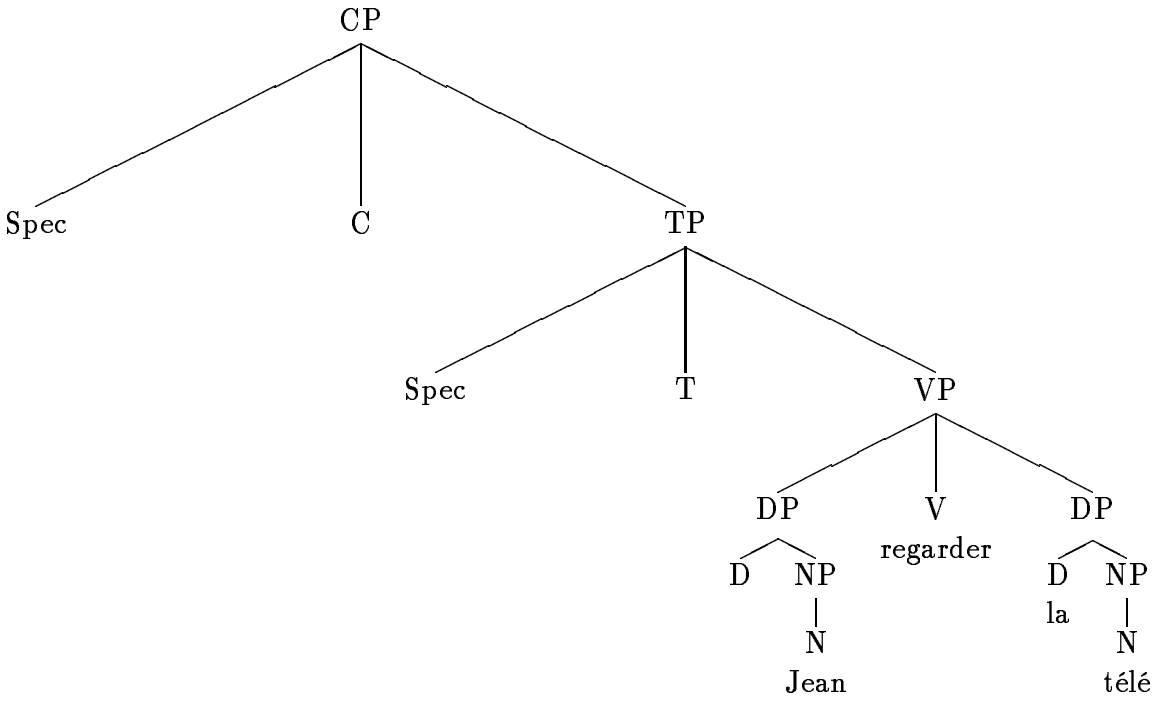
$\mathcal{M}$={[$_{\mathtt{CP}}$ C(∅)],[$_{\mathtt{TP}}$ T(présent)],[$_{\mathtt{VP}}$ V(regarder)], [$_{\mathtt{DP}}$ D(∅)],[$_{\mathtt{DP}}$ D(la)],[$_{\mathtt{NP}}$ N(Jean)],[$_{\mathtt{NP}}$ N(télé)] }

La troisième étape de la procédure entraîne à présent la construction finale de la D-structure. La tête C de la projection CP a pour propriété de sélectionner un TP; on insère donc le TP en Compl de CP. La tête T de TP sélectionne un VP, lequel est donc placé en Compl de TP. D sélectionne un NP, donnant une projection avec NP en Compl de DP.[6] Enfin, la tête V du VP assigne deux rôles thématiques, Agent et Thème, et sous-catégorise un DP. Le DP ayant le rôle Thème est placé en Compl de VP (satisfaisant

[6]Notons que l'on joint les DP et les NP en fonction également de la relation Opérateur-Propriété qui les lie dans la PSS.

la sous-catégorisation) et le DP Agent en Spec de VP. Ceci nous donne la représentation de D-structure en (69) :

(70)



Sur la base des représentations de D-structure nous allons pouvoir examiner à présent les opérations appliquées par le système GBGen.[7]

3.2 Génération syntaxique et définition des principes

La théorie GB est présentée (à l'instar de tous les modèles chomskyens) comme un modèle de *production* de phrases, partant des informations lexicales pour produire des phrases grammaticales. A première vue, l'implémentation de la théorie dans un générateur syntaxique devrait être relativement directe. En examinant plus précisément le fonctionnement de la théorie GB,

[7]Notons que dans ce qui suit les sujets ne seront pas générés en Spec de VP mais en Spec de IP.

il apparaît que le passage à l'implémentation des principes définis dans cette théorie n'est pas aussi immédiat. En effet, ces derniers sont vus dans le cadre linguistique GB comme des filtres éliminant les structures incorrectes. Or un système de génération automatique semble devoir plutôt être de type déterministe, au sens où un seul parcours dérivationnel est poursuivi jusqu'à obtention d'une phrase correcte, sans retour en arrière pour corriger des choix erronés. Les principes intégrés au générateur ne peuvent donc être vus comme des filtres. Dans les sections suivantes, nous illustrerons le traitement de certains principes fondamentaux de la théorie GB dans le générateur.

Pour l'essentiel nous conserverons l'idée de la théorie GB que les transformations qui permettent de passer de la D-structure à la S-structure se réduisent à une instruction de déplacement, *Déplacer-α*. Contrairement à l'optique couramment admise dans GB, cette instruction ne sera pas vue comme un instruction libre avec des principes filtrant les mouvements illicites, mais comme une opération permettant de *satisfaire* les contraintes de bonne formation des structures syntaxiques. Les sections suivantes illustrent et discutent cette approche quant aux deux grands types de mouvements reconnus dans la théorie GB, les mouvements A et A-barre.

3.3 Traitement du mouvement A-barre

Le terme de mouvement A-barre recouvre les déplacements vers une position A-barre, i.e. une position non-argumentale. Ce type de mouvement permet de former des interrogatives, des exclamatives et des relatives.[8] Nous nous limiterons ici au cas des interrogatives et aux relatives, que l'on peut voir comme les structures A-barre types.

Dans le cadre de la théorie GB, les constructions A-barre sont, dans la majorité des cas, formées par mouvement explicite d'un élément interrogatif (élément-wh désormais) de sa position de base (argument ou ajout) vers la position [Spec, CP], ce mouvement étant cyclique, i.e. l'élément-wh pouvant être déplacé de [Spec, CP] en [Spec, CP].[9] Les structures suivantes illustrent

[8] Cf. cependant Etchegoyhen (1997) pour des arguments contre une analyse A-barre des relatives.

[9] Nous n'aborderons pas ici le cas des interrogatives fermées (ou questions oui-non, du

ces déplacements d'argument (71a) ou d'ajout (71b), et le cas d'un mouvement cyclique (71c) :

(71)a. [$_{CP}$ Qui$_i$ as$_j$ [$_{TP}$ tu t$_j$ vu t$_i$]]

b. [$_{CP}$ Comment$_j$ viendras$_j$ [$_{TP}$ tu t$_j$ t$_i$]]

c. [$_{CP}$ Qui$_i$ crois$_j$ tu t$_j$ [$_{CP}$ t'$_i$ que [$_{TP}$ Marie aime t$_i$]]]

Le mouvement-wh ne se fait cependant pas sans contraintes: certains déplacements d'éléments-wh entraînent l'agrammaticalité de la phrase, (72a), une structure identique avec l'élément-wh in situ étant acceptable, avec un accent contrastif fort, (72b) :

(72)a. *Qui$_i$ aimes-tu l'idée que Marie aime t$_i$?

b. Tu aimes l'idée que Marie aime QUI ?

Ce type de données a été longuement étudié en grammaire générative, donnant lieu à des caractérisations précises des contraintes sur le mouvement-wh. Il est important de noter que tout générateur syntaxique doit faire face d'une façon ou d'une autre à ces contraintes sur la formation des interrogatives. L'option déterministe choisie pour notre générateur imposera une forme particulière à l'algorithme de traitement des constructions-wh. Les paragraphes suivants décrivent les différentes contraintes pesant sur les structures interrogatives, ainsi que les choix faits au niveau de l'implémentation.

Contraintes d'îlot, ECP et Contrainte de Supériorité Les travaux sur le mouvement-wh de (Ross J. 1967) ont mis en lumière ce qu'il est convenu d'appeler les Contraintes d'îlot, i.e. les contraintes sur l'extraction des éléments-wh. Nous centrerons l'analyse sur les contraintes suivantes :

• Contrainte sur les îlots-wh (WhC) : un élément-wh ne peut être extrait d'une subordonnée interrogative.

type *As-tu vu Jean ?*) qui n'impliquent pas de mouvement d'élément-wh lexical, bien qu'il soit probable qu'un opérateur vide soit déplacé dans ces constructions.

(73) * Qui$_i$ te demandes-tu quand$_j$ Marie a vu t$_i$ t$_j$

• Contrainte sur les structures coordonnées (CSC) : un élément-wh ne peut être extrait d'une structure coordonnée.[10]

(74) * Quel sport$_i$ aimes-tu [$_{\texttt{ConjP}}$ le vin et t$_i$]

• Contrainte du sujet (SC) : un élément-wh ne peut être extrait d'une catégorie (DP, CP) en position sujet.[11]

(75)a. *[$_{\texttt{CP}}$ où$_i$[$_{\texttt{CP}}$ PRO aller t$_i$] t'ennuies]

b. *[$_{\texttt{CP}}$ [de qui]$_i$ [$_{\texttt{DP}}$ le spectacle t$_i$] est annulé]

• Contrainte des DP complexes (CNPC) : un élément-wh ne peut être extrait d'un CP dominé par un DP.

(76)a. *[avec qui]$_i$ détestes-tu [$_{\texttt{DP}}$ l'idée [$_{\texttt{CP}}$ que Jean danse t$_i$]]

b. *[avec qui]$_i$ connais-tu [$_{\texttt{DP}}$ la personne [$_{\texttt{CP}}$ O$_i$ que Jean a vue t$_j$ t$_j$]]

[10]Notons que la contrainte interdit l'extraction hors d'une structure coordonnée, mais pas le mouvement de la structure coordonnée. Pour cela, tous les éléments coordonnés doivent être des éléments-wh, comme le montre le contraste suivant :
(i) [$_{\texttt{ConjP}}$ Quel sport et quel alcool]$_i$ aimes-tu t$_i$?
(ii) *[$_{\texttt{ConjP}}$ Quel sport et le vin]$_i$ aimes-tu t$_i$?

[11]L'existence d'une contrainte spécifique sur l'extraction hors d'un DP sujet laisse sous-entendre qu'une même extraction serait possible si ledit DP se trouvait en position d'objet. Nos jugements propres nous font cependant préférer le déplacement du DP entier (i) à l'extraction hors de ce DP (ii), dans certains cas :
(i) ? [De qui]$_i$ as-tu aimé le spectacle t$_i$
(ii) [Le spectacle de qui]$_i$ as-tu aimé ti ?
Nous considérerons donc par la suite que le déplacement du DP dans son entier est l'option préférable, ne tenant pas compte de ce fait de la contrainte sur l'extraction hors d'un DP sujet.

• Contrainte des PP (PPC) : un élément-wh ne peut être extrait d'un PP.

(77)a. *[De quelle mairie]$_i$ as-tu envoyé un pavé [$_{PP}$ sur le toit t_i]

b. *[De qui]$_i$ as-tu marché [$_{PP}$ sur la tête t_i]

A ces contraintes d'îlot s'ajoutent des contraintes comme le Principe des Catégories Vides (ECP). Ce dernier impose à toute trace d'être proprement gouvernée, c'est à dire, essentiellement, de se trouver dans un environnement local approprié. L'ensemble des cas couverts par l'ECP étant relativement vaste, nous nous bornerons ici à considérer le cas de l'extraction illicite des éléments-wh à partir de la position sujet d'une complétive tensée en français, comme illustré ci-dessous :

(78) * Qui$_i$ crois-tu [$_{CP}$ t'_i que [$_{TP}$ t_i déteste Marie]]

Cette structure viole l'ECP, la trace en position sujet de la subordonnée tensée n'étant pas proprement gouvernée.

Enfin, le générateur syntaxique devra intégrer une contrainte supplémentaire sur le mouvement-wh, appelée Condition de Supériorité. Cette dernière indique simplement que, dans le cas où plusieurs éléments-wh sont présents dans une même structure, seul l'élément dominant les autres (i.e. l'élément supérieur en termes structurels) est déplacé. Les données suivantes illustrent ce cas de figure :

(79)a. [$_{CP}$ Qui$_i$ [$_{TP}$ t_i a fait quoi]]

b. * [$_{CP}$ Quoi$_i$ [$_{TP}$ qui a fait t_i]]

La structure (79b) viole la condition de supériorité, l'objet étant déplacé et non l'élément-wh sujet qui le domine.

Principes linguistiques et contraintes computationnelles La question qui se pose lors de l'implémentation de ces contraintes sur le mouvement-wh est celle de l'adéquation entre théorie linguistique et algorithme de génération. Les contraintes d'îlot ont été, au cours de l'évolution de la théorie transformationnelle, remplacées par une contrainte plus générale, la Sous-jacence. Cette dernière empêche tout mouvement de franchir plus d'un noeud barrière en une seule étape (cf. Chomsky N.1986, notamment). Pour prendre un exemple concret, la CNPC, décrite plus haut, ne serait qu'une manifestation de la Sous-jacence, le mouvement de l'élément-wh en (80) traversant les deux barrières CP et DP en une seule étape :

(80) *[avec qui]$_i$ détestes-tu [$_{DP}$ l'idée [$_{CP}$ que Marie danse t_i]]

La Sous-jacence a été adoptée en remplacement des contraintes d'îlot du fait de sa plus grande généralité et de sa simplicité. Il semblerait donc souhaitable d'intégrer cette condition dans le générateur syntaxique. Cependant, on se heurterait alors à deux types de problèmes. Tout d'abord, se poserait un problème de traitement exhaustif des constructions-wh. Ainsi, la Sous-jacence traite difficilement le cas de l'extraction hors d'une phrase en position sujet (voir plus haut), l'élément-wh ne traversant qu'une barrière (CP) dans ce cas. Il serait donc nécessaire de projeter un noeud DP supplémentaire pour les phrases en position sujet, entraînant de fait des complications au niveau de l'algorithme de projection des structures-D. D'autre part, l'adéquation d'un tel traitement du point de vue de l'efficacité computationnelle ne va pas de soi. Une génération de surface basée sur la Sous-jacence pourrait prendre les formes suivantes :

A. L'arbre syntaxique est parcouru et pour chaque élément-wh rencontré on calcule le nombre de barrières intervenant entre cet élément et la première position [Spec, CP] accessible le dominant. Si ce nombre est supérieur à deux, la procédure ne déplace pas l'élément et cherche un autre élément-wh.

B. L'arbre syntaxique est parcouru et chaque noeud barrière est comptabilisé. Si un élément-wh est trouvé, le nombre de barrières intervenant entre cet élément et la position [Spec, CP] immédiatement dominante est calculé et la procédure suit le cours défini en A.

La procédure A implique une recherche de haut en bas pour chaque élément-wh rencontré, afin de trouver le nombre de barrières entre cet élément et la première position [Spec, CP] disponible, ce qui constitue une recherche couteuse qu'il conviendrait d'éviter. La procédure B est plus économique de ce point de vue. Toutes deux présentent cependant un désavantage commun, à savoir que les barrières seront toujours calculées. Dans le cas d'une interrogative enchâssée contenant plusieurs éléments-wh, ceci est inutile : dès lors qu'un élément-wh a été déplacé dans une telle configuration, il ne sera jamais possible dans déplacer un autre;[12] le calcul des barrières est inutile dans ce type de situation, le résultat dudit calcul étant toujours le même. Ceci s'applique également aux cas d'îlots forts, telles les phrases-ajout, dans lesquelles le déplacement-wh est illicite.

Il paraît donc plus efficace de définir un algorithme en posant notamment que la procédure ne peut s'appliquer qu'une seule fois dans le domaine d'une interrogative enchâssée, par exemple, évitant ainsi les calculs inutiles. L'algorithme de traitement des constructions-wh sera donc défini à partir des informations liées aux domaines locaux contenant les éléments-wh, comme détaillé à la section suivante. Ceci nous permettra d'intégrer dans une même procédure les contraintes d'îlot, l'ECP et la condition de supériorité.

Algorithme du mouvement-wh L'algorithme de traitement des constructions-wh comprend i) un parcours de l'arbre syntaxique, et ii) le mouvement-wh. La procédure générale de parcours de l'arbre prend en compte les informations associées à chaque domaine CP, dès qu'un noeud de ce type est rencontré. Ainsi, si dans le parcours de la structure un noeud CP marqué [+complément de nom] est rencontré, le mouvement-wh est inactivée jusqu'au prochain noeud CP, évitant ainsi de violer la CNPC appliquée aux phrases complément de nom. D'autre part, la procédure ne déplaçant que les éléments dits wh, il est nécessaire de définir la classe de ces éléments. Les éléments de ce type, susceptibles d'être déplacés, seront des projections maximales contenant un élément interrogatif. Plus précisément, une projection maximale sera marquée [+wh] (i.e. visible pour la procédure) sous certaines conditions. Ainsi, une structure coordonnée n'appartiendra à la classe des éléments-wh que si tous ses membres sont de type wh; dans le

[12] Du moins en français, cf. Rudin [1988] pour une étude sur les langues à déplacement-wh multiple.

cas contraire, la structure coordonnée ne sera pas dotée du trait [+wh] et ne sera donc pas affectée par la procédure, ce qui nous permettra de ne jamais violer la contrainte sur les structures coordonnées. De même, pour ne pas enfreindre les contraintes sur l'extraction hors d'un groupe prépositionnel ou d'un groupe nominal, le trait [+wh] est attribué à la projection DP ou PP maximale dominant l'élément-wh lexical, l'ensemble étant ainsi déplacé. Les interrogatives et les relatives seront également marquées d'un trait wh (noté wh(CP) dans l'algorithme). Enfin, les domaines enchâssés déclaratifs ne sont pas marqués [+wh], permettant ainsi la montée de l'élément-wh dans les domaines supérieurs, et sont désignés par complement(CP) dans l'algorithme.

Il est intéressant de noter que lorsque plusieurs éléments-wh apparaissent dans un même domaine CP, la traversée de l'arbre syntaxique étant descendante ("Top-down"), le premier élément-wh rencontré, et déplacé, est également l'élément supérieur, au sens de la condition de Supériorité décrite dans les sections précédentes. Cette condition est donc directement dérivée de la nature descendante de la procédure, un résultat important.

(81) Algorithme WH

- Parcourir la D-structure[13]
- Pour chaque premier élément-wh trouvé dans un domaine CP donné:
  - SI wh(CP) OU
    (complement(CP) ET
    toutes les positions [Spec, CP] dominantes sont vides[14] ET
    l'élément-wh n'est pas un sujet)
  - ALORS déplacer l'élément-wh jusqu'à la première position [Spec, CP] marquée [+wh].

[13]Le parcours est descendant récursif de gauche à droite (chaque descendante est poursuivie jusqu'au bout avant de traiter une autre branche de l'arbre). D'autre part, la descente part d'un CP (noeud local) et s'achève ou bien sur un terminal, ou alors sur un noeud CP, auquel cas la procédure s'applique récursivement à ce CP.

[14]La recherche de ces positions [Spec, CP] est peu coûteuse. La technique la plus simple consiste à utiliser une marque après un déplacement-wh dans ce type de position [Spec, CP]. Le parcours étant descendant, la marque permet de savoir immédiatement si les conditions pour le mouvement sont remplies.

Pour illustrer le fonctionnement de la procédure, considérons les structures profondes simplifiées suivantes:

(82)a. [$_{CP_{+WH}}$ qui se demander[$_{CP_{+WH}}$ Jean faire la vaisselle comment]]

b. [$_{CP_{+WH}}$ je me demander[$_{CP_{+WH}}$ qui aimer quoi]]

c. [$_{CP_{+WH}}$ la personne [$_{CP_{+WH}}$ je parler à qui] croire [$_{CP}$ que Marie vouloir [$_{CP}$ que Jean aimer qui]]]

d. [$_{CP_{+WH}}$ qui croit [$_{CP}$ que Jean voir qui]]

En (82a), la procédure parcourt tout d'abord la phrase matrice, trouve un élément-wh (*qui*), et le déplace en [Spec, CP]. La traversée de l'arbre se poursuit dans un nouveau domaine CP, qui a la propriété de ne pas être une principale mais d'être une subordonnée interrogative, et déplace donc l'adverbe wh (comment) vers la première position marquée [+wh], à savoir la position [Spec, CP] enchâssée.

En (82b), aucun élément-wh n'est trouvé dans le CP principal, la procédure passe donc au CP suivant, le complément enchâssé interrogatif, et déplace l'élément-wh supérieur, le sujet qui, vers la position [Spec, CP] enchâssée, première position marquée [+wh]. Notons que la position [Spec, CP] de la phrase matrice, bien que marquée [+wh], ne recevra aucun élément-wh, l'algorithme spécifiant que le mouvement-wh s'arrête au niveau du premier [Spec, CP] [+wh].

Dans la structure (82c), aucun élément-wh n'est rencontré dans la phrase matrice, la procédure traverse ensuite le CP de la relative, déplace l'élément-wh à qui vers la première projection [+wh] et traverse ensuite les autres domaines CP. L'élément-wh de la phrase la plus enchâssée est déplacé de [Spec, CP] en [Spec, CP] jusqu'à la position [+wh] de la phrase matrice.

Enfin, en (82d), l'algorithme déplace dans un premier temps l'élément-wh (qui) de la principale, trouve un deuxième élément de ce type dans la subordonnée, mais aucun déplacement n'affecte ce dernier, la position [Spec, CP] de la phrase matrice n'étant pas vide.

Les structures dérivées sont ainsi les suivantes (le choix précis du temps associé aux verbes n'est pas pertinent ici) :

(83)a. [$_{CP_{+WH}}$ qui$_i$ t$_i$ se demande[$_{CP_{+WH}}$ comment$_j$ Jean fait la vaisselle t$_j$]]

b. [$_{CP_{+WH}}$ je me demande[$_{CP_{+WH}}$ qui$_i$ t$_i$ aime quoi]]

c. [$_{CP_{+WH}}$ qui$_j$ est-ce que la personne [$_{CP_{+WH}}$ à qui$_i$ j'ai parlé t$_i$] croit [$_{CP}$ que Marie veut [$_{CP}$ que Jean aime t$_j$]]]

d. [$_{CP_{+WH}}$ qui$_i$ t$_i$ croit [$_{CP}$ que Jean a vu qui]]

Il convient de noter que certaines de ces phrases constituent des cas extrêmes, en ce sens qu'elles sont peu susceptibles d'être produites dans un texte donné. Ces structures montrent cependant que l'algorithme proposé ici permet de traiter des cas complexes où plusieurs éléments-wh apparaissent et où les contraintes sur le mouvement-wh sont en jeu.

3.4 Traitement du mouvement A

Le mouvement-A se distingue de sa contrepartie A-barre en ce qu'il déplace un élément d'une position A(rgumentale) vers une autre position A. Les constructions produites par mouvement-A sont notamment les constructions passives, (84a), inaccusatives, (84b), et à montée, (84c) :

(84)a. Le policier$_i$ sera condamné t$_i$

b. Jean t_i semble t_i détester les policiers

c. Les policiers$_i$ sont partis t_i

Le mouvement A est motivé par la recherche du Cas abstrait. En (84a), le DP *le policier* est généré dans la position de complément du participe *condamné*, lequel est défectif vis à vis de l'assignation du Cas Accusatif; le DP objet doit donc se déplacer pour recevoir un Cas, le Cas Nominatif assigné par T en l'occurence. En (84b), c'est la tête T de l'enchâssée qui ne peut assigner de Cas, les formes infinitives n'assignant pas le Cas Nominatif; le DP *Jean* est ici contraint de se déplacer en [Spec, TP] de la phrase matrice, où la forme finie de T assigne le Nominatif. Enfin en (84c) le verbe *partir* est un inaccusatif qui ne peut assigner un Cas structural à son complément *les policiers*, ce dernier devant se déplacer en [Spec, TP] pour recevoir le Cas Nominatif.[15]

Dans la théorie GB, des relations en cascade sont employées pour définir l'assignation du Cas : un Cas est assigné sous une relation de gouvernement, i.e. une relation de commande entre une tête lexicale et l'élément recevant le Cas.[16] Implémenter ces relations impliquerait un algorithme de traitement du Cas défini comme suit :

1. Rechercher tous les DPs en parcourant la D-structure.[17]
2. Pour chaque DP trouvé :
   (a) Chercher une tête lexicale appartenant à l'ensemble des gouverneurs potentiels,
   (b) Déterminer si la tête lexicale considérée gouverne le DP en cours de traitement :

[15] Cf. Belletti [1988] pour une analyse en termes de Cas structural vs. Cas inhérent, et Etchegoyhen & Tsoulas [1997] pour une approche différente de ces données.

[16] Nous ne détaillerons pas le type de commande utilisé, c-commande ou m-commande, cette notion n'étant pas pertinente pour le traitement du mouvement A que nous adopterons. Cf Barker & Pullum [1990] pour une définition générale des relations de commande.

[17] Le type de parcours de l'arbre syntaxique n'est pas directement pertinent ici.

i. Si oui, vérifier si la tête lexicale assigne un Cas (Nominatif, Accusatif, etc.). Si tel est le cas, reprendre à l'étape 1. Sinon reprendre à l'étape (a).

ii. Si non, reprendre à l'étape (a).

L'algorithme décrit ci-dessus est une version simplifiée d'un algorithme réel utilisant la notion de gouvernement. En effet, nous n'avons pas détaillé le calcul de la commande, ni les autres conditions entrant dans la définition du gouvernement (notamment les contraintes de localité que sont les Barrières et la Minimalité, cf. Rizzi 1990, parmi d'autres). La question qui se pose est la suivante : est-il nécessaire, dans le cas du mouvement A, d'implémenter un tel algorithme reprenant la formulation générale donnée dans la théorie GB? Comme pour le mouvement A-barre, nous opterons pour une réponse négative. L'algorithme suivant permet en effet de traiter les trois cas majeurs de mouvement A :

1. Dans chaque domaine CP, rechercher les DPs simples (non dominés par un PP) en [Spec, TP] et [Compl, VP],

2. Appliquer les conditions/actions suivantes :

   (a) Si le DP est en [Spec, TP] ET T non fini (infinitif), OU

   (b) Si le DP est en [Compl, VP] ET V a le trait [inaccusatif] ou [passif],

   (c) Déplacer le DP vers la première position [Spec, TP] avec T fini.

   (d) Sinon reprendre l'étape 1.

Avant de justifier un tel algorithme, voyons comment sont traités les trois grands cas de mouvement A. Le système produit les D-structures suivantes pour les exemple en (84) :

(85)a. [$_{CP}$ [$_{TP}$ être [$_{VP}$ condamné [$_{DP}$ le policier]]]]

b. [$_{CP}$ [$_{TP}$ sembler [$_{CP}$ [$_{TP}$ [$_{DP}$ Jean] [$_{VP}$ détester [$_{DP}$ les policiers]]]]]]

c. [$_{CP}$ [$_{TP}$ être [$_{VP}$ partis [$_{DP}$ les policiers]]]]

En (85a), le système ne trouve aucun DP en [Spec, TP] et un DP en [Compl, VP]. Ce dernier est dans le VP d'un verbe ayant le trait [passif], le mouvement s'applique donc, déplaçant le DP vers la première position [Spec, TP] avec T fini. En (85b), le système trouve un DP en [Spec, TP] de la phrase enchâssée, le DP *Jean* et, T étant non fini, déplace ce DP vers la position [Spec, TP] de la principale, où T est fini. Le deuxième DP trouvé dans cette structure, le DP *les policiers* en [Compl, VP] de la subordonnée, ne remplit aucune des conditions motivant un déplacement, ce DP est donc laissé en place. Ceci nous donne les structures de surface en (84).

L'algorithme ci-dessus est préférable du point de vue de la simplicité et produit les résultats attendus dans les trois cas donnés ci-dessus. Il est important cependant de le justifier d'un point de vue général.

Le premier point à justifier concerne la recherche locale de DPs dans les deux listes [Spec, TP] et [Compl, VP]. Dans le cadre du système, ces deux listes sont les seules susceptibles de contenir les DPs à D-structure. D'autre part, seuls les DPs peuvent devoir être déplacés par mouvement A; les DPs contenus dans des PPs recevant leur Cas de la préposition, vérifier leur statut Casuel est simplement inutile. La recherche peut donc, sans risques, se limiter aux DPs simples présents dans les deux listes mentionnées. Ceci posé, les cas de mouvement A se réduisent aux DPs simple compléments de verbes passifs ou inaccusatifs, et au DPs simples en [Spec, TP] où T est non fini. Dans ces domaines, la relation de gouvernement Casuel peut se réduire à la vérification des traits pertinents (finitude, passif, inaccusatif), la relation structurale de commande et la localité de la relation étant *dans tous les cas* établies entre un DP argument de V/T et ces têtes V/T. Il est donc là encore intuile de procéder à un calcul de ces relations (i.e. de la relation de gouvernement), calcul qui donnerait toujours un résultat positif.

Le traitement du mouvement que nous proposons doit également être justifié. Existe-t-il des cas où le déplacement d'un DP par mouvement A aboutit à une structure mal formée? Ce serait le cas par exemple s'il n'existait pas de position Casuelle disponible dans la structure. La théorie prédit de fait qu'une telle situation ne peut se présenter, étant donnée la généralisation de

Burzio, notamment, dont une partie est donnée ci-dessous :

Généralisation de Burzio : un verbe qui n'assigne pas de Cas à son objet n'assigne pas de θ-rôle à son sujet.

Ceci implique des configurations du type :

(86) [$_{\mathtt{TP}}$ T [$_{\mathtt{VP}}$ V DP]], où V n'assigne pas de Cas au DP objet.

En effet, si le verbe n'assigne pas de rôle externe, le système ne génère pas d'argument externe et la position [Spec, TP] sera disponible dans tous les cas où un objet ne peut recevoir de Cas. En d'autres termes, le mouvement A appliqué aux constructions passives et inaccusatives aboutira dans tous les cas au placement de l'objet dans la position Casuelle adéquate.

L'autre trait du système garantissant l'existence d'une position Casuelle pour un élément déplacé par l'algorithme du mouvement A est la définition même des verbes à montée. Ces derniers n'assignent en effet qu'un θ-rôle interne; comme précédemment, ceci garantit l'existence d'un site d'arrivée pour les DPs déplacés par mouvement A.

L'algorithme plus simple que nous proposons est donc en accord avec la contrainte de déterminisme placée sur le système GBGen. Comme l'algorithme du mouvement A-barre, il a également la particularité d'être plus simple que l'équivalent GB en termes de gouvernement, et donc d'être vraisemblablement plus efficace d'un point de vue computationnel.

3.5 En défense des choix : le cas de la D-structure

Dès les travaux fondateurs de la théorie GB, la question de l'existence de la D-structure a été posée. D'après Chomsky [1981], ce niveau peut être aisément éliminé de la grammaire, et remplacé par un mécanisme adéquat de formation directe des chaînes en S-structure. Un programme de ce type a du reste été entamé dans les années 80 par quelques chercheurs (cf. Rizzi

1986, notamment), et le programme minimaliste de Chomsky [1995] a totalement éliminé ce niveau de représentation syntaxique. L'idée de simplifier la grammaire, en réduisant le nombre de ses niveaux de représentation, est sans aucun doute attrayante du point de vue linguistique, et les résultats produits par les nouveaux modèles génératifs sans D-structure ne sont pas à négliger. Il peut donc paraître surprenant que nous ayons pris le parti de maintenir le niveau de D-structure pour le système GBGen. Dans cette section, nous nous attacherons à justifier brièvement ce choix, en prenant pour exemple l'implémentation des constructions interrogatives.

L'algorithme de dérivation des constructions interrogatives décrit précédemment a pour mérite sa grande simplicité et le fait de ne poser aucun problème majeur quant aux questions de déterminisme et d'efficacité. Imaginons cependant que le module de génération ne comporte pas ce niveau de D-structure. D'un point de vue très général, la procédure de formation des interrogatives dans un tel modèle devrait compter les étapes suivantes :

1. Détection d'un élément interrogatif dans la SPS.
2. Création d'une chaîne A-barre en syntaxe avec placement de l'élément-*wh* dans la position appropriée et insertion d'une trace dans la position canonique de cet élément.

Le système serait, en apparence, bien plus simple et élégant que celui proposé plus haut, qui compte une création de D-structure, un parcours descendant récursif de celle-ci et des procédures de déplacement. Un examen plus précis de l'option avec création de chaîne montre cependant qu'elle est difficilement tenable. En effet, comme on l'a vu à la section 3.2, les chaînes A-barre sont soumises à des contraintes fortes. Le pied d'une telle chaîne ne peut par exemple se trouver dans une proposition relative si la tête de la chaîne ne s'y trouve pas, comme illustré ci-dessous :

(87) *Qui$_i$ est-ce que Jean a vu la personne qui connaît t$_i$?

Dans l'optique de création de chaînes, l'élément-*wh* *qui* serait inséré en [Spec, CP] de la phrase matrice interrogative et le système chercherait à insérer la trace de cet élément pour compléter la chaîne. La position d'objet de connaître étant la seule position possible, la trace y sera placée, donnant

une structure agrammaticale. Il faudrait donc contraindre la formation des chaînes en filtrant certaines structures :

1. Vérifier la bonne formation de la chaîne. Si la chaîne est mal-formée, alors :
2. Détruire/effacer la chaîne, et
3. Créer une nouvelle chaîne en plaçant l'élément-*wh* dans sa position canonique.

Cette approche serait non-déterministe, impliquant une destruction de structure créée et une nouvelle opération de création de chaîne, un cas de figure impossible avec le traitement défendu plus haut. D'un point de vue général, les chaînes étant contraintes par les principes de la grammaire, les cas de destruction et recréation de chaînes seront fréquents et affaiblissent de ce fait l'intérêt d'une approche en termes de création directe de chaînes à S-structure, au profit d'une approche intégrant un niveau de D-structure et des mouvements contraints.

On pourrait objecter à ce raisonnement que les chaînes peuvent être contraintes en amont, c'est à dire avant la syntaxe. Dans une telle optique, le système examinerait les propriétés des CLS en examinant par exemple le statut d'îlot d'une CLS. Si un élément-*wh* est dans un îlot défini au niveau pseudo-sémantique, alors le système ne placerait pas cet élément en [Spec, CP] mais bien dans sa position de base. Il est évident que cette approche reviendrait à utiliser un parcours de la structure pseudo-sémantique similaire au parcours de la D-structure dans l'approche proposée plus haut, ce qui ne présenterait que peu d'intérêt. D'autre part, un problème plus important se poserait. Dans le cas d'une phrase contenant plusieurs éléments-*wh*, on a vu que seul l'élément structurellement supérieur devait être déplacé (Condition de supériorité). Une approche voulant contraindre la création des chaînes de façon déterministe, en examinant les informations pseudo-sémantiques, se verrait dans l'obligation d'établir des relations de dominance structurelle au niveau pseudo-sémantique, ce qui reviendrait encore une fois à faire de la pseudo-sémantique une D-structure. Il paraît donc peu souhaitable de définir un système créant directement des S-structures et des chaînes, et le modèle

utilisant des D-structures semble plus fondé du point de vue de la génération déterministe.[18]

3.6 Morphologie

Dans cette section nous donnerons un aperçu rapide du fonctionnement du module MORPH. Les phénomènes traités par celui-ci sont :

- La conjugaison (production des formes verbales fléchies),

  (88) être manger ⇒ est mangé

- Les accords (sujet-verbe, participe-passé),

  (89) Les enfants dormir ⇒ Les enfants dorment

- Les contractions (déterminant-nom, etc.)

  (90) Le enfant ⇒ L'enfant

- Les insertions (*il* explétif)

  (91) semble que Jean dort ⇒ Il semble que Jean dort

Les processus de conjugaison/accord sont relativement simples. Sur la base des informations de la PSS (nombre, personne), du lexique (genre), et de la configuration syntaxique (e.g., recherche du sujet pour l'accord verbal)

[18]Les points développés ici s'appliquent directement aux grammaires d'unification, lesquelles, n'intégrant pas de niveau équivalent à la D-structure mais des structures de surface produites directement, ne peuvent donner lieu à des systèmes de génération syntaxique déterministes. Reiter [1994] constate cependant que la plupart des composants de génération de surface basés sur des grammaires d'unification n'utilisent pas de retours en arrière ("backtracking"). Aucune autre précision sur le composant syntaxique des systèmes considérés par Reiter [1994] n'étant disponible, il ne nous est pas possible de discuter plus amplement cette question comparative. Ces systèmes font peut-être appel à du pseudo-parallélisme, ou bien les structures engendrées sont relativement simples, auquel cas les contraintes syntaxiques ne sont que peu pertinentes.

le système recherche dans la base de données lexicale la forme fléchie correspondant aux traits et insère cette forme à la place de la forme infinitive.

Il est un autre aspect de la génération qui doit être traité dans le module MORPH, c'est ce que l'on désignera par concordance des temps. Notons avant tout que ce que l'on appelle ainsi est finalement difficilement définissable (cf. Brunot [1936]). L'idée que la concordance des temps est un système contraignant ne permettant que peu de combinaisons temporelles s'avère fausse. Il apparaît en réalité que dans certains cas *toutes* les combinaisons sont possibles comme le montre le paradigme suivant avec un verbe déclaratif :

(92) Jean disait que Marie a regardé/avait regardé/regardait/regarderait/regarde/regardera la télé.

La séquence *Imparfait-Futur* peut sembler a priori choquante dans la phrase précédente. Il ne nous est cependant pas possible d'*exclure* une telle séquence, au vu des exemples suivants:

(93)a. Jean nous disait hier soir que Marie regardera la télé le jour où les poules auront des dents.

b. Jean racontait hier soir que la CIA mettra en place un programme d'élimination des groupes révolutionnaires dans le courant de l'année.

Ces phrases étant, à notre sens, parfaitement acceptables, nous n'excluerons pas *a priori* la séquence *Passé-Futur*. Le lecteur pourra vérifier par lui-même que, dans le cas de verbes principaux déclaratifs, aucune combinaison temporelle ne peut être totalement exclue. Cette affirmation doit être modulée. Il existe bel et bien ce que l'on pourrait appeler une concordance des *modes*. Ainsi, les verbes de souhait (*désirer, souhaiter)*, de doute (*douter, nier*) et de volonté (*vouloir, souhaiter*) interdisent l'indicatif dans la complétive qu'ils sélectionnent, imposant le subjonctif.[19] Ce fait entraîne une restriction dans le choix du temps de la subordonnée: le subjonctif ne

[19]Nous ne discuterons pas des langues comme l'allemand, qui utilisent l'indicatif dans ce type de cas.

comptant au maximum que quatre temps (passé, présent, imparfait, plus-que-parfait), la combinaison est de fait restreinte. Dans la pratique actuelle de la langue, hors emploi littéraire ou niveau de langue affecté, seuls le présent et le passé du subjonctif sont utilisés. La combinaison temporelle entre principale et subordonnée avec les verbes susmentionnés se réduit donc au choix du présent ou du passé du subjonctif dans la subordonnée. Lorsque le verbe principal est au passé ou au conditionnel, le verbe enchâssé devrait en théorie être au passé; l'usage actuel donne cependant une large préférence au présent du subjonctif dans ces cas là :

(94) Je voulais qu'il vînt/vienne

Lorsque le temps de la principale est le présent ou le futur, le temps de la subordonnée est toujours le présent :

(95) Je veux/voudrai qu'il vienne.

Par souci de simplification, et dans le registre de langue non soutenu que nous adoptons pour l'instant dans le cadre de GBGen, les verbes imposant le mode subjonctif en français à leur complétive imposeront également le présent à la phrase enchâssée.[20] Il y a deux possibilités de traitement de ce type de contrainte : i) au niveau de la PSS, le système analyse le type de verbe de la PSS principale, et assigne les valeurs Real ou Unreal pour le type Color dans la subordonnée, ou bien ii) le module MORPH examine les propriétés du verbe principal et engendre une forme indicative ou subjonctive dans la complétive. La première option semble préférable, le subjonctif exprimant une valeur sémantique différant de l'indicatif (en français, cf. la note 15) il paraît plus logique d'établir la distinction au niveau pseudo-sémantique. Le module de morphologie n'aurait ainsi qu'à rechercher la forme correcte du subjonctif ou de l'indicatif.

Penchons nous à présent sur le cas des ajouts phrastiques. Les modes/temps possibles en français sont les suivants, label par label :[21]

[20]Nous considérons ici le cas de propositions subordonnées fléchies, il va de soi qu'une subordonnée infinite est possible avec tous les verbes décrits ci-dessus.

[21]Dans les cas où le mode subjonctif est possible, nous n'indiquons pas systématiquement la distinction présent/passé. Le critère simple établissant une connexion entre le temps (présent/passé) de la principale et le temps de la phrase ajout se révèle en effet inadéquat, cf *Il venait sans que je l'appelle.* Il y a cependant quelques cas clairs où l'une des deux options est impossible, cf. le cas de 6.

1. (WITHOUT | CLS) : Subjonctif/Infinitif.

   (96) J'ai détruit la maison blanche sans **qu'on ne m'aie vu/qu'on ne me voie/faire de bruit**

2. (FOR | CLS) : Subjonctif présent/infinitif.

   (97) Je ferai mon travail **pour que mon chef me paye/changer le monde**

3. (MODU | CLS) : Subjonctif.[22]

   (98) Je travaille selon **que les gens le demandent ou non/que les gens l'aient demandé ou non**

4. (MODU_OPP | CLS) : Subjonctif.

   (99) Je dors **bien que la révolution mondiale ait éclaté/que la révolution gronde autour de nous**

5. ([-duration]/E_P<E_Sem | CLS) : Subjonctif/Infinitif.

   (100) Je fuirai **avant que la police ne m'attrape/que la police ne m'aie attrapé/de finir en prison**

6. ([-duration]/E_P>E_Sem | CLS) : Indicatif/Infinitif. [23]

   (101) Je reviendrai chez moi **après que la police est partie/*que la police parte/avoir achevé ma tâche**

[22] Nous laissons de côté le cas marginal de l'emploi du futur, comme dans **Je ferai mon travail selon qu'il me sera payé ou non.**

[23] Notons que toute structure infinitive ne serait pas appropriée. Il conviendrait de rajouter l'information aspectuelle d'accomplissement qui seule convient dans ce cas. Signalons cependant que l'information aspectuelle pourrait être la bonne caractérisation abstraite associée à ces labels; c'est le cas pour ce label-ci, le subjonctif passé et l'infinitif avec *avoir* exprimant l'accomplissement. Nous nous contenterons ici de la réalisation concrète du label en français, avec le mode correspondant.

7. ([-duration]/E_P=E_Sem | CLS) : Temps de l'indicatif équivalent au temps de l'indicatif de la principale.[24]

   (102) Je dors **quand je veux/Il parlait quand tout le monde se taisait/Jean viendra quand les autres partiront**.

8. (INTERVAL/E_P<E_MOD | CLS) : Indicatif présent ou passé.

   (103) Je mange mieux depuis **qu'on m'a augmenté/qu'on me traite convenablement**

9. (OPP | CLS) : $T_{principale}$.

   (104) Elle sortait **alors qu'il pleuvait**/Elle sort **alors qu'il pleut**/Elle sortira **alors que ses amis l'attendront**

10. (LINK | CLS) : $T_{principale}$[25]

    (105) Je ne serai pas tranquille **tant qu'ils seront au pouvoir**/Rien ne changeait **tant que les mandarins régnaient**/On est prisonniers **tant que le système ne change pas**

11. (HYP | CLS) :

    - $T_{principale}$=futur/présent (indicatif) $\Rightarrow$ présent

      (106) Je viens/viendrai **si tu me le demandes**

    - $T_{principale}$=conditionnel présent $\Rightarrow$ imparfait (indicatif)

      (107) Ils chanteraient **si on le leur demandait**

    - $T_{principale}$=conditionnel passé $\Rightarrow$ plus-que-parfait (indicatif)

[24] Il est des exemples trompeurs ici, du type *Il partait à la plage quand Marie arriva*, où les temps ne sont pas identiques. Ceci est dû à l'ambiguité aspectuelle du français. Lorsque l'imparfait est utilisé dans le sens progressif, l'ajout introduit par *quand* est au passé simple; si l'imparfait n'est pas utilisé avec ce sens aspectuel, la concordance des temps s'applique, donnant *Il partait quand Marie arrivait.*

[25] Lorsque le temps de la principale est le futur, le présent est également acceptable dans la phrase ajout, cf. *Je ne serai pas tranquille tant qu'ils sont au pouvoir.*

(108) **S'ils avaient été mieux payés**, ils auraient été plus productifs

- $T_{principale}$=Impératif ⇒ présent (indicatif).

  (109) Fais-le, **si tu l'oses**

12. (CAUSE1 | CLS) : $T_{principale}$=passé ⇒ Même temps; $T_{principale}\neq$passé ⇒ Indicatif présent.

    (110) Il mangeait **parce qu'il n'avait pas le choix**/Il a mangé **parce qu'il n'a pas eu le choix**/Il mangera **parce qu'il le veut bien**

13. (CAUSE2 | CLS) : $T_{principale}$=passé ⇒ Indicatif passé; $T_{principale}\neq$passé ⇒ Indicatif présent.

    (111) Je buvais, **puisque tu ne voulais pas m'augmenter/Puisque c'est comme ça**, je rentre chez ma mère

Les relations décrites ici seront gérées par le système au niveau pseudo-sémantiques, suivant une procédure générale du type :

- Pour chaque PSS dépendante d'une PSS principale,
  - Si la PSS dépendante est dans la liste d'arguments de la PSS principale (PSS_P), vérifier si le V de PSS_P est de type souhait, doute ou volonté, et attribuer la valeur Unreal à la PSS dépendante si la condition est vérifiée.
  - Si la PSS dépendante est dans la liste de modifieurs de PSS_P, vérifier le type de SemanticLabel et assigner une valeur autorisée (cf. la liste ci-dessus).

Le type de calcul sur les PSS décrit ici est relativement simple, et permettra au module MORPH d'aller rechercher les formes verbales appropriées dans la base de données lexicale.

Chapter 4

Conclusion

4.1 Remarques finales

Ce mémoire avait pour but principal de présenter le système de génération syntaxique GBGen et les principes théoriques à la base du projet. Nous avons tenté de donner une idée aussi claire que possible des grands composants du système, de la représentation des connaissances, les structures pseudo-sémantiques, à la génération syntaxique proprement dite.

Il est important de noter que la façon de traiter des problèmes particuliers peut dépendre de l'application même dans laquelle est placée le générateur. Par exemple, le traitement de la sélection des modes verbaux dans les phrases ajout ou complément peut être différent suivant que les structures pseudo-sémantiques sont créées par un utilisateur ou dérivées d'une structure syntaxique préexistante.[1] Dans la mesure du possible, nous avons essayé de nous abstraire de cette question, ce qui a pu dans certains cas donner lieu à une formulation des procédures très générale.

Ceci étant, nous avons donné les grands traits d'une représentation des connaissances pseudo-sémantique qui permet de fournir l'information suffisante au fonctionnement du générateur syntaxique. Cette base du système a été définie de façon à pouvoir servir dans un cadre de traduction automatique, en autorisant notamment la représentation d'entités non phrastiques.

[1] Ceci correspond aux deux programmes GENE et TIPS, cf. le chapitre introductif.

Ces représentations seront naturellement modifiées et/ou étendues au cours des recherches futures.

Le point fondamental que nous souhaitons avoir illustré concerne la façon dont les contraintes pesant sur la formation des structures syntaxiques peuvent être implémentées dans un système de génération déterministe. La théorie GB, dans sa variante incluant un niveau de D-structure, permet d'établir des procédures relativement simples et générales aboutissant à des structures de surface bien formées. Ceci a été illustré dans le traitement des deux grands types de mouvement, A et A-barre. Il va de soi que la classe des phénomènes syntaxiques est loin d'avoir été traitée ici. Il n'est qu'à citer le cas des pronoms, des extrapositions, de la focalisation, entre autres, pour se convaincre de l'ampleur de la tâche à accomplir dans l'élaboration d'un système à large échelle de génération syntaxique.

Le système GBGen n'a cependant *a priori* pas de limitation intrinsèque enpêchant son extension à la couverture d'autres phénomènes syntaxiques, en français ou dans d'autres langues. Pour donner une illustration concrète de l'extension possible du générateur, il serait relativement simple de traiter les déplacements-wh multiples dans la formation des interrogatives dans certaines langues (cf. Rudin 1988), en appliquant l'algorithme du mouvement-wh plus d'une fois par domaine CP. Ceci reviendrait à placer un paramètre sur le nombre d'applications du mouvement-wh suivant les langues. L'idée de la théorie GB, également appelée Théorie des Principes et Paramètres, serait ainsi conservée : sur la base d'un algorithme général (e.g., l'algorithme du mouvement-wh), des paramètres permettraient de rendre compte de la variation entre langues. D'autres paramètres pourraient aisément être définis pour créer des D-structures à ordre SOV, par exemple.

La recherche en génération syntaxique ne doit pas, à notre sens, être vue comme un affaiblissement de la théorie linguistique. Bien que les phénomènes traités par le système soient loin de couvrir l'ensemble des phénomènes décrits dans les travaux en linguistique, il nous semble que l'élaboration de générateurs automatiques prouvant leur efficacité dans la production de phrases permettra de donner une nouvelle orientation à la recherche en linguistique. Ainsi, contraindre la grammaire avec l'option déterministe impose une redéfinition des principes linguistiques pouvant être profitable à la linguistique.

Pour donner une illustration rapide, le fait de dériver la Condition de Supériorité des mécanismes de recherche descendants (cf. la section 3.3) peut être vu comme un résultat *linguistique* important. Il reste à développer des procédures générales de production de phrases, tout à la fois efficaces et couvrant les cas mis à jour par la recherche en linguistique proprement dite.

Enfin, notons que l'option déterministe nous a permis d'élaborer un système dont la caractéristique première est vraisemblablement sa très grande efficacité. Ce simple constat nous incite à pousser plus avant le développement du système, en conservant les options théoriques majeures présentées dans ce mémoire.

4.2 Références

Appelt, D. E. [1985]. *Planning English Sentences*, Cambridge University Press.

Balicco, L. [1993]. *Génération de répliques en français dans une interface homme-machine en langue naturelle*, thèse, Université Pierre Mendès France, Grenoble.

Barker, Ch. & Pullum, G. [1990]. "A Theory of Command Relations", Linguistics and Philosophy, 13 : 1-34.

Barwise J. et Cooper R. 1981. "Generalized quantifiers and natural language", Linguistics and Philosophy, 4 :159-219.

Belletti, A. [1988]. "The Case of Unaccusatives", Linguistic Inquiry, vol.19-1 : 1-34.

Block, R. [1988]. "Can a parsing grammar be used for natural language generation? The negative example of LFG", *Advances in Natural Language Generation*, Zock M. & Sabah G. (eds), Pinter Publishers, London.

Brunot, F. [1936]. *La pensée et la langue*, Paris.

Chomsky, N. [1955].*The Logical Structure of Linguistic Theory*, University of Chicago Press (1985).

Chomsky, N. [1981]. *Lectures on Government and Binding*, Dordrecht, Foris.

Chomsky, N. [1986]. *Barriers*, Cambridge, MIT Press.

Chomsky, N. [1995]. *The Minimalist Program*, MIT Press.

Clark, R. [1993]. "Semantics for Computers", ms. LATL, Université de Genève.

Clark, R. & Wehrli, E. [1995]. "Natural language processing, lexicon and semantics", Methods of Information in Medicine, 34.1.

Comrie, B. [1981]. "On Reichenbach's Approach to Tense", CLS 17, 24-30.

Dale, R. & Reiter E. [1995]. Computational Interpretations of the Gricean Maxims in the generation of referring Expressions", *Cognitive Science*, vol. 19, 2, 233-263.

Danlos, L. [1985]. *Génération Automatiques de Textes en Langues Naturelles*, Paris, Masson.

Danlos, L. [1992]. "Contraintes syntaxiques et pronominalisation en génération de textes" in Anis (éd.) 36-62.

Dorr, B. [1990]. *Lexical Conceptual Structure and Machine Translation*, doctoral dissertation, MIT, Cambridge, MA.

Dorr, B. & Gaasterland, T. [1995]. "Selecting Tense, Aspect and Connecting Words in Language Generation", *Proceedings of IJCAI-95*, Montreal, Canada.

Dorr, B. & Broman Olsen M. [1996]. "Multilingual Generation: the Role of Telicity in Lexical Choice and Syntactic Realization", *Journal of Machine Translation*, 11:1-3.

Elhadad, M. & Robin, J. [1996]. "An Overview of SURGE: a Reusable Comprehensive Syntactic Realization Component", Proceedings of NLGW, Brighton.

Etchegoyhen, T. [1997]. "Against *wh*-movement in relative clauses", à paraître dans les Actes du colloque Langues et Grammaire III, Paris.

Etchegoyhen, T. & Pinnagoda, S. [1997]."Un algorithme pour la génération déterministe de constructions interrogatives", actes de GAT'97, Grenoble, France.

Etchegoyhen, T. & Tsoulas, G. [1997]. "Theticity & Definite Descriptions. A Case Study.", à paraître dans *Romance Linguistics: theoretical Perspectives*, Current Issues in Linguistic Theory, John Benjamins.

Friedman, J. [1971]. *A Computer Model of Transformational Grammar*, New York, Elsevier.

Haegeman, L. [1991]. *Introduction to Government and Binding*, Oxford, Basil Blackwell. Nouvelle édition révisée et augmentée, 1994.

Iordanskaja, L., R. Kittredge et A. Polguère, [1988]. "Implementing the Meaning-Text model for language generation", Coling-88, Budapest.

Iordanskaja, L., R. Kittredge et A. Polguère. [1991]. "Lexical selection and paraphrase in a Meaning-Text generation model", in Paris, C., W. Swartout et W. Mann (éd.), 293-312.

Joshi, A. [1987]. "The relevance of tree adjoining grammar to generation", in Kempen G. (éd.) *Natural Language Generation*, 233-52, Dordrecht, Martinus Nijhoff Publishers.

Keenan E., Stavi J. [1986]. *A semantic characterization of natural language determiners*, Linguistics and Philosophy, 9 :253-326.

Korelsky, T. et O. Rambow. [1992]. "Applied text generation", *ACL Proceedings of the Third Conference on Applied Natural Language Processing, Trento*, 40-7, Association for Computational Linguistics.

Laenzlinger, C. et E. Wehrli, [1991]. "FIPS : Un analyseur interactif pour le français", *TA Informations*, 32:2, 35-49.

Larson, R. K. [1988]. "On the Double-Object Construction", Linguistic Inquiry, 19 : 335-391.

Matiasek J. & Trost, H. [1996]. "An HPSG-based Generator for German", *Proceedings of COLING-96.*

Matthiessen, C. et J. Bateman. [1992]. *Text Generation and Systemic-Functional Linguistics*, Londres, Pinter Publishers.

McDonald D. et L. Bolc (éd.) [1988]. *Natural Language Generation Systems*, New York, Springer Verlag.

Mellish C. et R. Evans. [1989]. "Natural language generation from plans", *Computational Linguistics*, 15.5, 233-249.

Merlo, P. [1993]. "For an Incremental Computation of Intrasentential Coreference", actes de IJCAII-93 : 1216-1221.

Nogier, J-F. [1991]. *Génération automatique de langage et graphes conceptuels*, Paris, Hermès.

Paris, C., W. Swartout et W. Mann (éds.) [1990]. *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, Dordrecht, Kluwer Academic Publishers.

Pullum, G.K. [1986]. "Assuming Some Version of X-bar Theory", actes de CLS 86.

Reichenbach, H. [1947]. *Elements of Symbolic Logic*, Free Press, New York.

Reiter, E. [1994]. "Has a Consensus NLG Architecture Appeared, and is it Psycholinguistically Plausible?", actes de ENLGW-94.

Rizzi, L. [1986]. "On Chain formation", *The Grammar of Pronominal Clitics*, Syntax and Semantics, 19. New York: Academic Press.

Rizzi, L. [1990]. *Relativized Minimality*, MIT Press.

Ross, J. [1967]. *Constraints on Variables in Syntax*, Thèse de Doctorat, MIT.

Rudin, C. [1988]. "On Multiple Questions and Multiple Wh-fronting *Natural Language and Linguistic Theory*, vol. 6, 4.

Saint-Dizier, P. [1989]. " A generation method based on principles of government and binding theory" *Proceedings of the 2nd European NL generation workshop*, Edinburgh.

Shank, R. (ed) 1975.*Conceptual Information Processing*, North, Holland.

Shieber, S., G. van Noord, F. Pereira et R. Moore. 1990. "Semantic head-driven generation", *Computational Linguistics*, 16.1, 30-42.

Shieber, S. et Y. Shabes. 1990. "Generation and synchronous tree-adjoining grammars", in McKeown K., J. Moore et S. Nirenburg (éd.) *Actes du 5e colloque international de génération de langage*, Dawson, Pennsylvania.

Stowell, T. [1981]. *Origins of Phrase Structure*, thèse de doctorat, MIT.

Tsoulas, G. [1997]. "Sets and Syntax", Ms. Université de York, UK.

van Noord, G. 1990. "An overview of head driven bottom-up generation" in Dale et al. (1990).

Williams, E. [1986]. "A Reassignement of the Functions of LF", Linguistic Inquiry 17, 265-299.

Zock, M. et G. Sabah (éds) 1988. *Advances in Natural Language Generation: an Interdisciplinary Perspective*, Londres, Pinter, & Norwood, Ablex.

Zock, M. et G. Sabah 1992. "La génération automatique de textes : trente ans déjà, ou presque" in Anis (éd.) 1992, 8-35.